

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

David Možina

**Argumentirano strojno učenje z
uporabo logistične regresije**

MAGISTRSKO DELO
MAGISTRSKI PROGRAM DRUGE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: doc. dr. Matej Guid

Ljubljana, 2017

AVTORSKE PRAVICE. Rezultati magistrskega dela so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko Univerze v Ljubljani. Za objavljane ali izkoriščanje rezultatov magistrskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

©2017 DAVID MOŽINA

ZAHVALA

Iskreno se zahvaljujem mentorju doc. dr. Mateju Guidu za napotke in nasvete pri nastanku dela. Rad bi se zahvalil tudi staršem, ki so me spodbujali v času študija. Posebna zahvala gre tudi Heleni Milavec za pomoč pri izdelavi magistrskega dela.

David Možina, 2017

Kazalo

Povzetek

Abstract

1	Uvod	1
1.1	Motivacija	3
1.2	Predstavitev teme in cilji	4
1.3	Pregled dela	6
2	Nadzorovano strojno učenje in logistična regresija	7
2.1	Nadzorovano strojno učenje	7
2.2	Logistična regresija	8
2.3	Pregled sorodnih del	14
3	Argumentirano strojno učenje	17
3.1	Zanka za zajem ekspertnega znanja	18
3.2	Pregled sorodnih del	23
4	Logistična regresija z možnostjo argumentiranja	25
4.1	Izbira in izdelava novih atributov za napoved	27
4.2	Učenje hipoteze na učnih podatkih	28
4.3	Iskanje kritičnega primera	28
4.4	Razlaga kritičnega primera	28
4.5	Argumentiranje	29
4.6	Iskanje protiprimerov	30

KAZALO

4.7	Pridobivanje znanja iz argumentov	31
5	Grafični vmesnik	35
5.1	Okno za izbor podatkov	35
5.2	Okno za izbor atributov	36
5.3	Okno za argumentiranje primerov	37
6	Evalvacija	41
6.1	Mere za evalvacijo	41
6.2	Vpliv novih atributov na napovedno točnost	43
6.3	Zajemanje znanja iz eksperta	47
7	Sklepne ugotovitve	65

Seznam uporabljenih kratic

kratica	angleško	slovensko
CA	classification accuracy	napovedna točnost
ROC	receiver operating characteristic	ROC krivulja
AUC	area under curve	površina pod krivuljo ROC
ABML	argument-based machine learning	argumentirano strojno učenje
SVM	support vector machine	metoda podpornih vektorjev
LR	logistic regression	logistična regresija
GUI	graphical user interface	grafični vmesnik

Povzetek

Ljudje dandanes pri soočanju z novimi podatki vedno bolj stremimo k uporabi enostavnih orodij in postopkov za njihovo obdelavo in analizo. Želimo si na preprost način izluščiti uporabne informacije, odkriti zakonitosti v podatkih ali izdelati napovedni model.

V magistrskem delu smo zasnovali orodje, ki na enostaven način omogoča spoznavanje novih podatkov ter enostavno vpeljavo ekspertnega znanja v domeno v obliki novih značilk oz. atributov. Orodje temelji na paradigmi argumentiranega strojnega učenja (angl. *argument-based machine learning*, *ABML*) in metodi strojnega učenja, ki jo poznamo pod imenom logistična regresija. Modelu logistične regresije smo dodali možnost zajema ekspertnega znanja in razvili novo metodo, ki omogoča interakcijo med domenskim strokovnjakom (ekspertom) in logistično regresijo. Poimenovali smo jo argumentirano strojno učenje z uporabo logistične regresije (angl. *argument-based machine learning with logistic regression*).

Implementirali smo aplikacijo z grafičnim vmesnikom, ki uporablja omejeno novo metodo za zajem ekspertnega znanja. Metoda s pomočjo interaktivne zanke v gradnjo napovednega modela vključi dodatno znanje v obliki novih značilk oz. atributov. Poišče problematične primere, ki jih logistična regresija uvrsti v napačen razred. Ti primeri so predstavljeni domenskemu strokovnjaku oz. ekspertu. Ekspertova naloga je, da s podajanjem čim boljših argumentov v interakciji s programom razloži kritične primere in poda razloge, čemu določeni problematični primer sodi v nasprotni razred. Glede na podane argumente strokovnjaka metoda samodejno poišče relevan-

tne protiprimere, ki ekspertu olajšajo argumentiranje in včasih opozorijo na možne pomanjkljivosti v podanih argumentih. Protiprimeri se menjavajo sproti glede na navedene pogoje eksperta. Strokovnjak z vnašanjem domenskega znanja usmerja algoritem strojnega učenja do boljših napovednih točnosti in do modela, ki je skladen z ekspertnim znanjem.

Aplikacija poleg uporabe omenjene metode med samim postopkom argumentiranja omogoča tudi tvorjenje novih atributov. V primeru, da ti uspešno rešijo trenutni kritični primer, se ta zamenja, kar omogoča hitrejšo interakcijo med domenskim ekspertom in metodo strojnega učenja.

Ključne besede

umetna inteligenca, strojno učenje, znanost o podatkih, logistična regresija, argumentirano strojno učenje, logistična regresija z argumenti, interaktivna zanka za zajemanje znanja

Abstract

People nowadays tend to use simple tools and procedures to process and analyze new data. We are eager to find easy solutions to extract useful information from it and build predictive models.

In this thesis we designed a tool, which can easily be used to cope with a new data and which also enables a possibility to articulate expert's domain knowledge in the form of new features, i.e. attributes. The tool is based on a paradigm of *argument-based machine learning* (ABML) and machine learning method, called logistic regression. We modified the logistic regression method by adding the possibility to articulate the expert's domain knowledge and developed a new method, that allows interaction between domain expert and logistic regression called *argument-based machine learning with logistic regression*.

We created an application with a graphical user interface that uses newly created method and, by using interactive loop, captures domain expert's knowledge. The knowledge is passed into the predictive model in the form of new attributes. Method searches for problematic examples which are examples that are wrongly predicted by a logistic regression model. These examples are presented to the domain expert. Expert's task is now to explain critical examples by giving arguments and providing explanations for wrongly predicted example. According to the given expert's arguments, method finds relevant counterexamples which can highlight possible flaws and shortcomings in the expert's arguments. Counterexamples change regularly, based on the conditions mentioned by the expert. The interaction between the expert

and argument-based machine learning method can lead to better and more accurate models that are consistent with expert's domain knowledge.

The newly created application also enables creating new attributes, which can be made during the argumentation process. If the newly created attribute solves current critical example, it gets replaced by a new problematic example. This leads to a faster interaction between a domain expert and the machine learning algorithm.

Keywords

artificial intelligence, machine learning, data science, logistic regression, argument-based machine learning, argument-based logistic regression, knowledge refinement loop

Poglavje 1

Uvod

Danes se srečujemo s hitro rastočo količino podatkov. Zasluge za to gredo predvsem tehnologiji, ki se hitro razvija, internetu, ki omogoča priklop različnih vrst naprav na svetovni splet, in samim napravam, ki beležijo različne vrste podatkov. Tukaj govorimo o podatkih senzorjev na pametnih napravah, ki danes niso več nobena redkost, dnevniških datotekah, ki beležijo različne dostope do sistema in nenavadno oz. nepričakovano delovanje sistema, pametnih hišah, ki nam omogočajo kontrolo in nadzor nad delovanjem objekta na daljavo, avtomobilih, ki nas opozarjajo na nepravilnosti ali nevarnosti, in drugih pametnih napravah. Različna podjetja take vrste podatkov shranjujejo že vrsto let, saj jih lahko s primerno obdelavo in znanjem spremenijo v konkurenčno prednost.

Čuti se potreba po enostavnih pristopih in orodjih za obdelavo in analizo podatkov, s katerimi bi bilo možno izluščiti uporabne informacije. Postopkom, ki rešujejo določen problem v računalništvu pravimo algoritmi. Algoritem predstavlja zaporedje ukazov, ki se izvršijo in nas pripeljejo do rešitve problema. Tak primer je npr. urejanje zaporedja števil po velikosti od najmanjše do največje. Za rešitev tega problema obstaja več vrst različnih algoritmov, mi pa se odločimo, katerega bomo uporabili – npr. tistega, ki porabi najmanj ukazov, najhitrejšega ali katerega drugega [1].

Obstajajo pa tudi druge vrste nalog, za katere izbira algoritma ni samou-

mevna. Eden izmed primerov je način razvrščanje elektronske pošte. Tu nas pogosto zanima, ali gre za zeleno ali neželjeno elektronsko pošto. Algoritmu kot vhodni parameter podamo pošto, ki je pravkar prispela na naš spletni naslov, od algoritma pa si želimo dobiti povratno informacijo, za katero vrsto pošte gre – zeleno ali neželjeno. Ker se spletna pošta skozi čas spreminja in razlikuje od posameznika do posameznika, bi želeli imeti algoritem, ki te spremembe upošteva in se iz njih uči. Poiskati želimo pravila, ki se pojavijo v podatkih in s katerimi bi računalnik znal pravilno napovedati izhodno vrednost spremenljivke. Temu procesu pravimo strojno učenje. Osnovni princip strojnega učenja je opisovanje podatkov, rezultat tega pa so lahko pravila, funkcije, enačbe, ipd. Rezultate strojnega učenja uporabimo pri gradnji napovednega modela, ki bo znal napovedati izhodno vrednost spremenljivke na še ne videnih podatkih.

Algoritme strojnega učenja delimo na naslednje tri kategorije:

- nadzorovano strojno učenje (angl. *supervised machine learning*),
- nenadzorovano strojno učenje (angl. *unsupervised machine learning*),
- delno nadzorovano učenje (angl. *semi-supervised learning*).

Razlika med navedenimi kategorijami strojnega učenja je še zlasti v podatkih, ki jih imamo na voljo. Pri nadzorovanem strojnem učenju poznamo neodvisne spremenljivke (atribute ali značilke) in njihove odvisne spremenljivke (oznake ali razrede). Attribute in razrede v učni množici podatkov uporabimo za izdelavo funkcije, ki bo znala preslikati testne (oz. nepoznane) primere v izhodno spremenljivko. Pri nenadzorovanem strojnem učenju odvisnih spremenljivk ne poznamo. Cilj nenadzorovanega strojnega učenja je najti skrite vzorce v podatkih (npr. pri odkrivanju skupin sorodnih primerov), uporaben pa je tudi za predpripravo ali obdelavo podatkov pred postopkom nadzorovanega strojnega učenja. Delno nadzorovano strojno učenje sodi nekam vmes. Pri delno nadzorovanem učenju poznamo odvisne spremenljivke samo za določen del učnih primerov [2].

Nadzorovano strojno učenje bolj podrobno razdelimo na dve različni napovedni tehniki, ki se razlikujeta glede na tip odvisne spremenljivke. To sta:

- uvrščanje ali klasifikacija (angl. *classification*) in
- regresija (angl. *regression*).

Pri klasifikaciji so kategorije, ki jih napovedujemo, tipično diskretne številke, pri regresiji pa napovedujemo dejansko vrednost številke. Primer klasifikacije je prej omenjena detekcija elektronske pošte na zeleno in neželeno pošto. Za vhodne podatke uporabimo prejeto elektronsko pošto, za razred oz. klasifikacijo pa, ali gre za zeleno ali neželeno pošto. Med regresijske probleme sodi npr. napovedovanje cene določenega izdelka, kjer nas zanima napoved točne cene izdelka [3].

1.1 Motivacija

Algoritmi strojnega učenja vse bolj postajajo del računalniške programske opreme. Njihova uporaba sega od spletnega iskalnika, ki nam po pomembnosti razvršča spletne strani, do aplikacij za samodejno prepoznavanje obrazov na slikah, odkrivanja nepričakovanega delovanja sistema, prepoznavanja goljufij (angl. *fraud detection*), zaznave govora, samodejne vožnje avtomobila, klasifikacije elektronske pošte, priporočilnih sistemov za izboljšanje uporabniške izkušnje pri nakupovanju in še bi lahko naštevali. Posledično se čuti porast potrebe po kadru z analitičnim znanjem ter po novih idejah in rešitvah, ki bi pripomogle k boljšim napovedim algoritmov.

Srečavamo se s problemom, kjer smo soočeni z novimi podatki, o katerih ne vemo nič ali zelo malo in jih zato ne znamo spremeniti v konkurenčo prednost. Primanjkuje nam orodij, ki bi temeljila na spoznavanju podatkov, hkrati pa bi radi na enostaven način odkrili relacije med podatki in zgradili napovedne modele, v katere bi lahko vpeljali nova znanja (v obliki sestavljenih, tipično kompleksnejših atributov), ki bi izboljšali napovedno

točnost tovrstnih modelov. Eno izmed takšnih orodij je orodje *Orange* [4], vendar ne nudi ustreznega grafičnega vmesnika, kar bi omogočalo enostavno interakcijo med metodo strojnega učenja in strokovnjakom, ki je značilna za argumentirano strojno učenje.

V magistrski nalogi bomo predstavili orodje, ki omogoča interakcijo med algoritmom logistične regresije in domenskim strokovnjakom. Navedeno orodje med drugim omogoča, da se avtomatsko poiščejo mejni oz. problematični primeri, ki jih metoda strojnega učenja uvrsti v napačen razred in zna, glede na vnešene argumente strokovnjaka, poiskati ustrezne protiprimere, ki ne sovpadajo z razlago eksperta, s tem pa mu omogoča oz. olajša izboljšanje argumentov.

1.2 Predstavitev teme in cilji

V magistrski nalogi se bomo usmerili na algoritem nadzorovanega strojnega učenja – klasifikacijo oz. uvrščanje. Klasifikacijski model se na učnih podatkih nauči pravil, ki bodo znala preslikati testne podatke v enega izmed podanih razredov. Eden takšnih modelov je tudi logistična regresija. Gre za enostaven algoritem, ki v praksi dobro deluje. Uporabna je v domenah z atributi številskih vrednost $X = \{x_1, x_2, \dots, x_n\}$ in binarnim ciljnim razredom y . Metoda vrača verjetnost ciljnega razreda $y = 1$, ki ga izračuna s pomočjo logistične funkcije (1.1). Logistična funkcija kot vhodni parameter dobi uteženo vrednost parametrov modela in atributov (1.2). Algoritem logistične regresije bomo povezali z domenskim strokovnjakom (v nadaljevanju tudi ekspert), ki mu bo preko aplikacije omogočen vnos domenskega znanja v model.

$$p(y | X) = \frac{1}{1 + e^{-f(X)}} \quad (1.1)$$

$$f(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n \quad (1.2)$$

V zgornjo funkcijo (1.2) z dodatnim atributom x_i in parametrom modela θ_i enostavno vpeljemo domensko znanje eksperta. Ekspert svoje znanje poda v obliki pravil, ki jih pretvorimo v nov atribut in dodamo modelu logistične regresije. Možnost vpeljave argumentiranega strojnega učenja v metodo logistične regresije je prvi omenil Martin Možina v svoji doktorski disertaciji z naslovom "Argumentirano strojno učenje" [5]. Z vpeljavo interakcije med domenskimi eksperti in metodo logistične regresije dobimo novo metodo, ki upošteva argumente eksperta. Nova metoda pri gradnji napovednih modelov omogoča upoštevanje ekspertnih argumentov, s tem pa vodi do napovednih modelov, ki imajo praviloma večjo napovedno točnost in so tudi bolj skladni z ekspertnim znanjem.

V magistrski nalogi predlagamo novo metodo, ki bo temeljila na kombinaciji argumentiranega strojnega učenja (*ABML*) in algoritmu logistične regresije. Novo metodo smo izdelali tako, da s pomočjo interaktivne zanke za zajemanje ekspertnega znanja, ki temelji na strojnem učenju s pomočjo logistične regresije, v gradnjo napovednega modela vključimo dodatno znanje v obliki novih atributov. Strokovnjak glede na podan kritičen primer ob pomoči protiprimerov v algoritmu vnaša svoje domensko znanje. Ponujeni so mu robni primeri, ki jih razloži in s tem algoritmom usmerja k bolj točnemu napovedovanju, hkrati pa mu aplikacija služi kot orodje za učenje in spoznavanje podatkov. Podobno metodo, ki omogoča argumentiranje primerov v domenah nenadzorovanega strojnega učenja, je predstavil Peter Šaponja v svojem magistrskem delu z naslovom "Odkrivanje skupin s pomočjo argumentiranega strojnega učenja" [6]. Razvil je metodo, ki domenskemu strokovnjaku omogoča interakcijo z algoritmom *k-means* za odkrivanje skupin. Matevž Pavlič je v svojem magistrskem delu z naslovom "Ocenjevanje kvalitete argumentov pri argumentiranem strojnem učenju" zasnoval tri pristope za podajanje takojšnje povratne informacije o kakovosti argumenta [7]. Enega izmed teh pristopov bomo za isti namen uporabili tudi mi.

1.3 Pregled dela

Poglavji 2 in 3 vsebujeta nadaljevanje uvoda v magistrsko delo. Glavni prispevki magistrskega dela so predstavljeni v poglavjih 4 in 5. Rezultate predstavimo v poglavju 6. Delo zaključimo s sklepnimi ugotovitvami v poglavju 7.

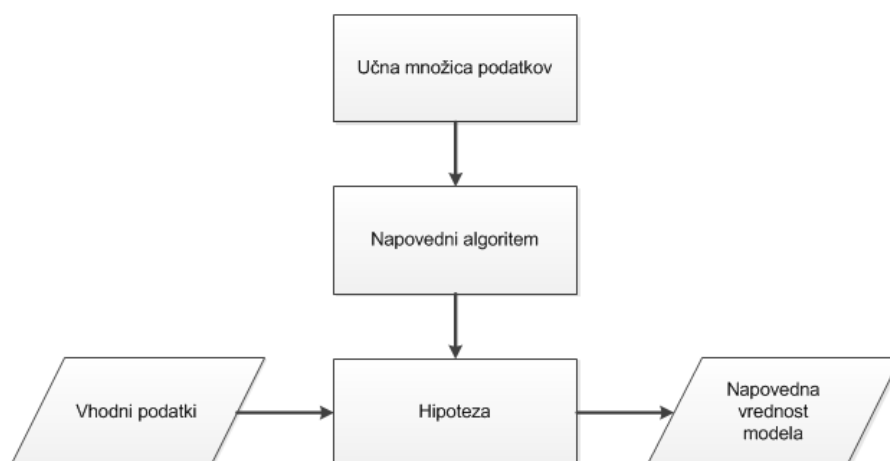
- V poglavju 2 opišemo metodo strojnega učenja logistično regresijo in predstavimo njeno delovanje, ter naredimo kratek pregled sorodnih del, povezanih z možnostjo vplivanja ekspertnih argumentov na učenje napovednih modelov.
- V poglavju 3 predstavimo paradigmo argumentiranega strojnega učenja, predstavimo potek interaktivne zanke za zajemanje ekspertnega znanja in naredimo kratek pregled sorodnih del, povezanih z argumentiranim strojnim učenjem.
- V poglavju 4 se osredotočimo na izdelavo metode, ki bo povezovala algoritem logistične regresije z metodo argumentiranega strojnega učenja ABML in predstavimo posamezne dele postopka, ki jih novo izdelana metoda uporablja za zajem ekspertnega znanja.
- V poglavju 5 predstavimo aplikacijo z grafičnim vmesnikom, ki smo jo razvili z namenom omogočanja oz. olajšanja interakcije z domenskim ekspertom. Aplikacija temelji na novo razviti metodi.
- V poglavju 6 predstavimo rezultate in delovanje novo izdelane metode ponazorimo na primerih.
- V poglavju 7 delo zaključimo s sklepnimi ugotovitvami in možnostmi za izboljšave.

Poglavje 2

Nadzorovano strojno učenje in logistična regresija

2.1 Nadzorovano strojno učenje

Algoritem za nadzorovano strojno učenje rešuje probleme, kjer je naš cilj na učni množici podatkov se naučiti hipotezo $\mathbf{h} : X \rightarrow Y$, ki bo znala za vhodne testne podatke (X) ustrezno napovedati vrednost izhodne spremenljivke (y). Prikaz postopka za pridobitev hipoteze $h_{\theta}(x)$ je prikazan na sliki 2.1.



Slika 2.1: Potek nadzorovanega strojnega učenja.

Hipoteza je funkcija, ki kot vhodni parameter dobi nov primer podatkov in vrne napoved približne vrednosti izhodne spremenljivke [3].

2.2 Logistična regresija

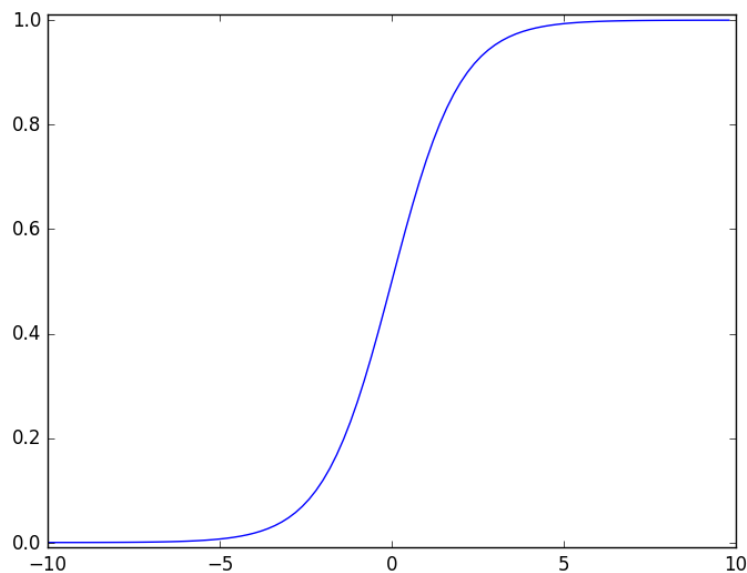
Logistična regresija je algoritem, ki rešuje problem klasifikacije in spada v kategorijo nadzorovanega strojnega učenja. Danes velja za enega od najbolj uporabljenih in priljubljenih algoritmov za klasifikacijo. Osnovni model klasifikacije ima dva diskretna razreda, ki ju označimo z dvema vrednostima odvisne spremenljivke $y = 0$ (*negativni razred*) in $y = 1$ (*pozitivni razred*).

Pri logistični regresiji bi radi našli hipotezo (angl. *hypothesis*), ki nam bo vračala vrednosti med 0 in 1. Ker regresija vrača poljubne vrednosti odvisne spremenljivke, moramo poiskati način, kako bi te vrednosti pretvorili v razred. Za dosego tega cilja bomo uporabili logistično funkcijo (rečemo ji tudi sigmoidna funkcija). Ta funkcija vrača vrednosti v intervalu med 0 in 1. Logistična funkcija se na pozitivni strani približuje vrednosti 1, na negativni strani pa vrednosti 0, kot je prikazano na sliki 2.2. Vrednosti, ki jih vrača logistična funkcija, bomo interpretirali kot verjetnost, da bo odvisna spremenljivka sodila v razred $y = 1$. Verjetnost za razred $y = 0$ enostavno izračunamo s formulo za verjetnost nasprotnega dogodka $P(y = 0) + P(y = 1) = 1$ [3, 8, 9].

2.2.1 Delovanje logistične regresije

Pri logistični regresiji gre za nadzorovano strojno učenje, kjer je cilj na učnih podatkih naučiti se povezavo med atributi in ciljnim razredom. Ključna predpostavka je, da obstaja linearna povezava med atributi in razredom. Za vsak opazovan primer so podani atributi, ki jih vzamemo kot vektor vhodnih vrednosti in izhodno vrednost, ki je podana kot razred. Predpostavimo, da imamo primer, ki vsebuje tri attribute. Attribute označimo z x_1, x_2, x_3 .

$$X^\top = [x_1, x_2, x_3]$$



Slika 2.2: Sigmoidna funkcija.

Vektorju X^\top na začetku dodamo še dodaten atribut x_0 , ki predstavlja vrednost odmika. Vrednost odmika privzeto nastavimo na 1 in dobimo nov vektor vhodnih vrednosti.

$$X^\top = [1, x_1, x_2, x_3]$$

Predpostavljamo, da lahko opredelimo izhodni razred y kot utežene vsote teh treh atributov in dodatnega atributa, pomnoženega z vektorjem parametrov modela θ^\top . Postopek izračuna vektorja parametrov modela bo predstavljen v nadaljevanju. Definirajmo naslednje enačbe:

$$\theta^\top x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n \quad (2.1)$$

$$h_\theta(x) = g(\theta^\top x) \quad (2.2)$$

$$g(z) = \frac{1}{1 + e^{-z}} \quad (2.3)$$

Kadar bo $\theta^\top x \geq 0$, bo primer klasificiran v razred $y = 1$, drugače pa bo klasificiran v razred $y = 0$. Dobljeno vrednost $\theta^\top x$ vstavimo v logistično funkcijo (2.3), da dobimo verjetnost za dogodek, da bo odvisna spremenljivka $y = 1$. Ta verjetnost je poljubno realno število med 0 in 1. Funkcija $h_\theta(x)$ (2.2) nam za vhodne podatke vrne verjetnost npr. 0,7. Zapišemo $h_\theta(x) = 0,7$. To pomeni, da vrednost 0,7 predstavlja višjo verjetnost razreda $y = 1$. Za izhodno spremenljivko $y = 0$ verjetnost izračunamo po formuli za verjetnost nasprotnega dogodka in dobimo 0,3. Rezultat, ki je trenutno izražen kot verjetnost, je potrebno spremeniti v razred. Za doseg tega cilja moramo določiti mejo, ki bo verjetnosti spremenila v razred. Najbolj enostavno mejo za določanje lahko postavimo kar na sredino, torej na 0,5. Za verjetnosti $h_\theta(x) \geq 0,5$ bi model klasificiral v razred 1, verjetnosti $h_\theta(x) < 0,5$ pa v razred 0 [3, 9].

2.2.2 Odločitvena meja za določanje razreda

Iz vrednosti, ki jih dobimo z logistično funkcijo, bi radi določili razrede. Odločitveno mejo (angl. *decision boundary*) za določanje razreda bomo za prikaz delovanja postavili na 0,5. Napoved razreda bo $y = 1$ v primeru, da bo funkcija $h_\theta(x)$ vrnila verjetnost večjo ali enako 0,5, in razred $y = 0$ v primeru, da bo verjetnost manjša od 0,5.

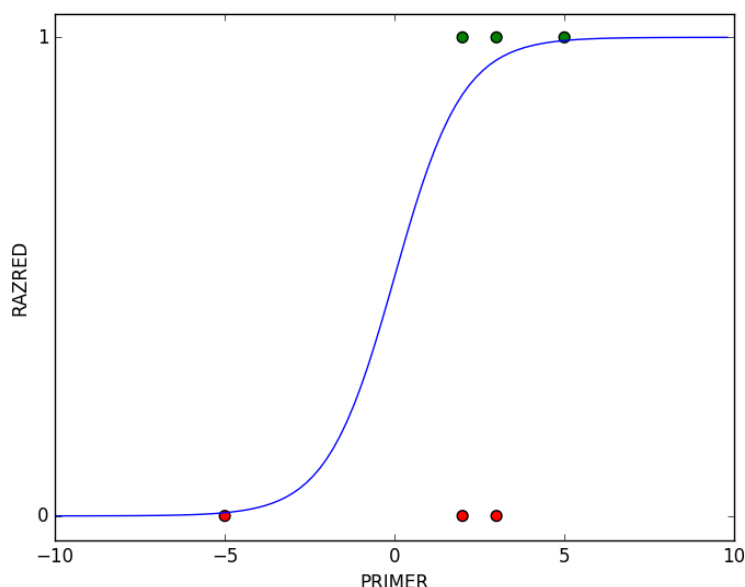
Poglejmo si delovanje na primeru. Vektor, ki predstavlja uteži parametrov modela dobljenih s pomočjo modela logistične regresije in podatkov iz učne množice, naj bo $\theta^\top = [-1, 3, -4]$. Prva številka (-1) v vektorju θ^\top predstavlja vrednost odmika, druga številka (3) predstavlja vrednosti uteži za prvi atribut, tretja številka (-4) pa vrednost uteži za drugi atribut.

Sedaj vzemimo primer iz testne množice podatkov, ki ima vrednost prvega atributa $x_1 = 2$ in vrednost drugega atributa $x_2 = 3$. Za ta primer bi radi izračunali vrednost diskretnega razreda y . Attribute predstavimo kot vektor števil. Vektorju dodamo tudi dodaten atribut x_0 z vrednostjo $x_0 = 1$.

Dobimo vektor $x_1 = [1, 2, 3]$. Po enačbi (2.1) naredimo skalarni produkt med vektorjem parametrov modela θ^\top in vektorjem atributov (X) prikazano v (2.4).

$$\theta^\top x = [-1, 3, -4] \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = -1 + 6 - 12 = -7 \quad (2.4)$$

Dobimo vrednost -7 . To vrednost vstavimo v logistično funkcijo (2.3) in dobimo verjetnost za razred $y = 1$, ki je 0. Ker je dobljena verjetnost 0 manjša od 0,5, primer klasificiramo v razred $y = 0$. Poglejmo si primer za vektor $x_2 = [1, 2, 1]$. Po zgornjem postopku dobimo verjetnost 0,73, ki je večja od meje 0,5. Primer klasificiramo v razred $y = 1$. Dodamo še nekaj primerov: $x_3 = [1, 5, 0.5]$, $x_4 = [1, 3, 2.5]$, $x_5 = [1, -5, 1]$ in $x_6 = [1, 3, 1.9]$ in točke izrišemo na grafu. Z rdečo barvo na sliki 2.3 so pobarvane točke v razredu $y = 0$, z zeleno barvo pa točke, ki smo jih klasificirali v razred $y = 1$. Za izris točk na abscistni osi smo vzeli drugo številko iz posameznega vektorja. Izrisana je tudi modra črta, ki predstavlja logistično funkcijo.



Slika 2.3: Klasifikacija primerov v razred.

Ko se bo pojavil nov primer, zopet vzamemo parametre modela θ^T in jih s skalarnim produktom pomnožimo z vektorjem atributov, ki mu prej dodamo vrednost odmika $x_0 = 1$. Rezultat skalarnega produkta vstavimo v logistično funkcijo in dobimo verjetje za razred $y = 1$. Glede na postavljeno mejo določimo razred. V primeru, da se v podatkih pojavi nov atribut, se parametri za model ponovno izračunajo [3, 9].

2.2.3 Določanje parametrov modela – kriterijska funkcija

V logistični regresiji s pomočjo kriterijske funkcije (angl. *cost function*) poiščemo parametre modela, ki se najbolj prilegajo podatkom. Za namen iskanja parametrov modela je uporabljen gradient v optimizaciji. Kriterijska funkcija se določi glede na odstopanja med napovedmi modela in dejanskimi podatki. Za prikaz delovanja kriterijske funkcije definirajmo spodnje formule:

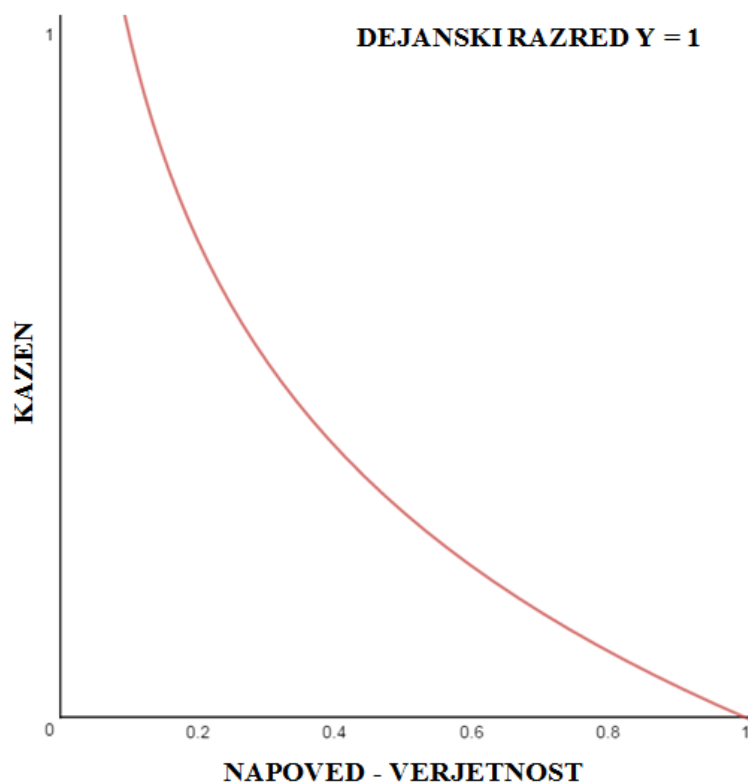
$$J(\theta) = \frac{1}{n} \sum_{i=1}^n \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)}) \quad (2.5)$$

$$\text{Cost}(h_{\theta}(x), y) = -\log(h_{\theta}(x)) \quad (2.6)$$

$$\text{Cost}(h_{\theta}(x), y) = -\log(1 - h_{\theta}(x)) \quad (2.7)$$

Kriterijska funkcija za razred $y = 1$ je podana z enačbo (2.6), za razred $y = 0$ pa z enačbo (2.7). Za lažjo predstavitev obe funkciji prikažimo na sliki.

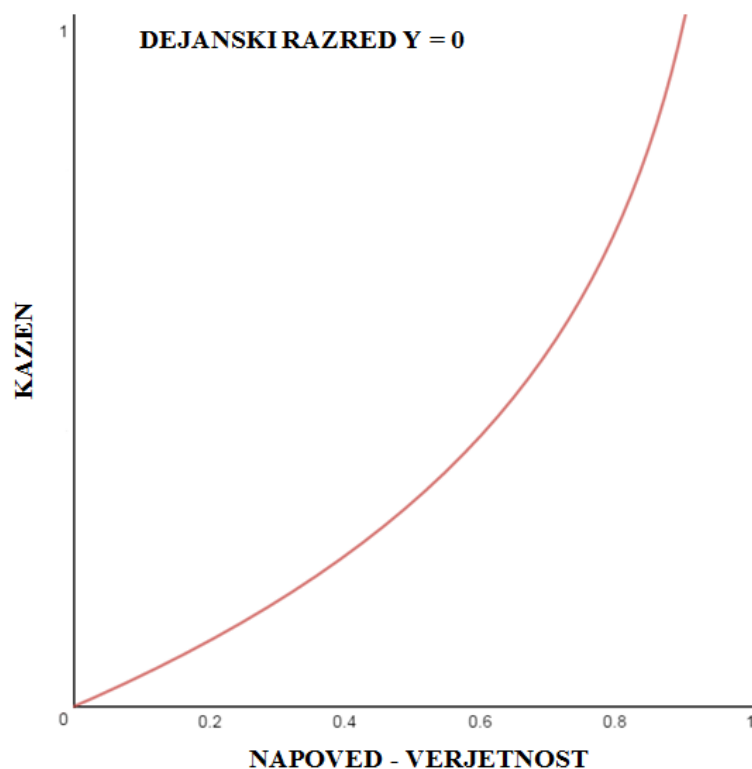
Iz slike 2.4 opazimo, da ima funkcija nekaj dobrih lastnosti. Kazen za pravilno napoved je 0. To je takrat, ko je dejanski razred $y = 1$ in je napovedana verjetnost za razred $h_{\theta}(x) = 1$. Obratno je pri napačni napovedi. Takrat, ko je dejanski razred $y = 1$ in napovedni model napove verjetnost, ki se približuje 0, bi želeli imeti čim večjo kazen. Ker gre za popolnoma napačno napoved, bi radi čim bolj kaznovali algoritem, kar zgornja funkcija tudi omogoča, saj se pri vrednosti $h_{\theta}(x) = 0$ kazen približuje neskončnosti.



Slika 2.4: Kriterijska funkcija za dejanski razred $y = 1$.

Poglejmo še kriterijsko funkcijo za razred $y = 0$, prikazano na sliki 2.5. Pri razredu $y = 0$ bi radi bolj kaznovali napovedi, ki jih algoritem napačno napove oz. jih napove z visoko verjetnostjo. Obratno je pri pravilni napovedi. V primeru, da je razred $y = 0$ in je napovedana verjetnost nizka, si ne želimo kaznovati algoritma oz. ga želimo kaznovati z minimalno kaznijo [9, 10].

Delovanje kriterijske funkcije najlažje pokažemo na primeru: model logistične regresije napove, da je 95 % možnosti, da elektronska pošta, ki smo jo prejeli, sodi v razred zelene pošte, v resnici pa prejeta elektronska pošta sodi med neželjeno pošto. Ker se je model občutno zmotil pri napovedi, uporabimo ta primer in mu dodelimo visoko vrednost kazni (večje, kot je odstopanje od dejanskega razreda, večja bo kazen).

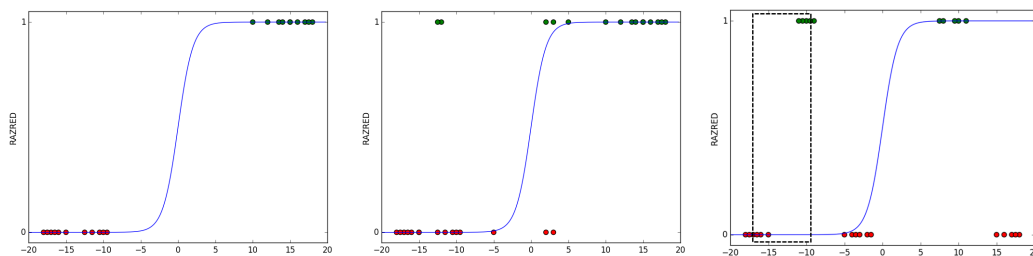


Slika 2.5: Kriterijska funkcija za dejanski razred $y = 0$.

2.3 Pregled sorodnih del

Logistična regresija je preprosta linearna metoda, kar je razlog, da jo lahko enostavno spreminjamo in ji dodajamo nove funkcionalnosti.

Sama po sebi pa ima tudi določene pomanjkljivosti. Glavna pomanjkljivost je ta, da ne deluje dobro nad nelinearnimi podatki oz. nad podatki, ki imajo več različnih segmentov (glej sliko 2.6 (desno)). Osnovni model logistične regresije deluje dobro nad preprostimi vzorci (glej sliko 2.6 (levo)) in vzorci, ki vsebujejo malo šuma v podatkih (glej sliko 2.6 (v sredini)). To pomanjkljivost lahko rešimo tako, da uporabimo metodo, ki smiselno razdeli primere v več prostorov oz. segmentov in se na vsakem prostoru naučimo svojo logistično funkcijo oz. model za napoved. Druga možna rešitev je uporaba lokalno utežene logistične regresije, pri kateri utežimo parametre



Slika 2.6: Različni vzorci podatkov. Slika levo prikazuje preproste vzorce podatkov, slika v sredini vzorce z malo šuma v podatkih, slika desno pa zapleten vzorec podatkov.

modela. Tretja rešitev, ki smo jo izdelali v sklopu magistrskega dela, pa omogoča tvorjenje novih relevantnih atributov.

Ena izmed idej, kako smiselno razdeliti podatke v več segmentov, je predstavljena v [11]. Predstavili so metodo, ki so jo poimenovali LMT (angl. *logistic model tree*) in je kombinacija metode induktivnih dreves (angl. *tree induction*) ter metode logistične regresije. Metoda rekurzivno razdeli primere v več prostorov, dokler se v listih dreves ne pojavi večina primerov z enakim razredom.

Druga možnost je uporaba lokalno utežene logistične regresije, ki jo je v svoji doktorski disertaciji omenil Kan D [12]. Glavni namen metode lokalno utežene logistične regresije je utežiti parametre modela θ glede na vrednost atributa. Pri lokalno uteženi logistični regresiji se vsak posamezen parameter θ_i spreminja glede na vrednost atributa x , pri logistični regresiji pa je parameter modela vedno enak. Predpostavljamo, da je v primeru dveh podobnih atributov x_1 in x_2 podoben tudi njun pripadajoči parameter modela θ_1 oz. θ_2 .

Novo razvita metoda olajša tvorjenje relevantnih atributov s pomočjo avtomatske detekcije problematičnih primerov, s pomočjo protiprimerov pa pomaga odpravljati pomanjkljivost v ekspertovih oz. uporabnikovih razlagah teh primerov. Nove attribute ustvarimo z osnovnimi matematičnimi operacijami iz originalne množice atributov. Pri izdelavi atributov lahko uporabimo tudi novo izdelane attribute. Z novimi atributi se logistična regresija nauči

kompleksnejših mej za določanje razreda. Izdelava novih atributov je pogosto rešitev do boljšega napovednega modela [13]. V [14] avtorji izdelavo novih atributov navajajo kot ključ do zmage na priznanem tekmovanju.

Sorodna dela, povezana z argumentiranim strojnim učenjem, so obravnavana v naslednjem poglavju.

Poglavje 3

Argumentirano strojno učenje

Argumentirano strojno učenje (angl. *argument-based machine learning, ABML*) je metoda, ki omogoča zajemanja znanja iz domenskega strokovnjaka. Sestavljena je iz dveh delov:

- spremenjenega algoritma za strojno učenje, ki upošteva argumente eksperta, in
- interaktivne ABML zanke, ki ekspertu omogoča vnos svojega znanja v algoritem.

Združuje koncepte strojnega učenja in argumentiranja. Običajno strojno učenje se nad učnimi podatki nauči npr. določenih pravil, ki pa pogosto niso smiselna z vidika domenskega strokovnjaka, ki domeno dobro pozna. Pri uporabi argumentiranega strojnega učenja domenskemu strokovnjaku ponudimo možnost, da z vnašanjem svojega znanja v model vpliva na iskanje hipoteze. Metoda domenskemu ekspertu ponudi robne primere, ki so podani z vektorjem atributov in razredu, ki mu določen primer pripada. Ekspert primer razloži in s tem usmerja algoritem k napovedim, skladnim s svojim domenskim znanjem. Z argumenti ekspert pojasni, čemu podani primer sodi v določen razred. Ločimo *pozitivne* in *negativne* argumente, ki so odvisni od razreda, ki mu primer pripada. Primere, ki jih argumentiramo, imenujemo *argumentirani primeri* [15, 16].

Tabela 3.1: Prednosti ABML

Lastnost	Prednost
razlaga le enega primera hkrati	olajšano podajanje znanja
kritični primeri	ekspert razlaga le relevantne primere, možnost odkrivanja morebitnih napak v podatkih (npr. napačno podan razred)
protiprimeri	odkrivanje pomanjkljivih argumentov
dodajanje novih atributov	modeli skladni z domenskim znanjem, višja napovedna točnost

Glavne prednosti argumentiranega strojnega učenja so povzete v tabeli 3.1.

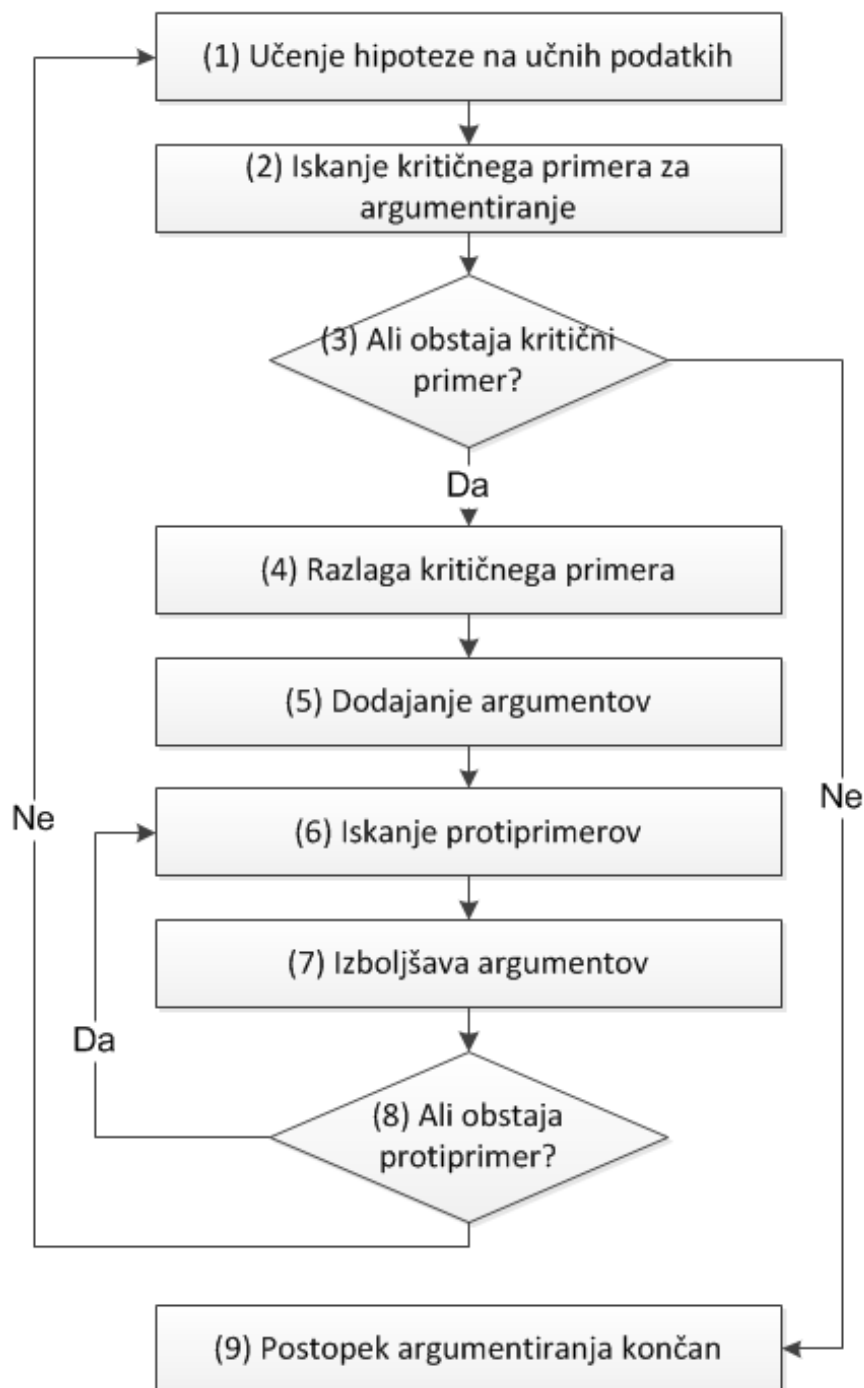
3.1 Zanka za zajem ekspertnega znanja

Interaktivna zanka za zajemanje ekspertnega znanja (angl. *knowledge refinement loop*) je pomemben del algoritma ABML. Gre za metodo, ki jo ekspert uporablja pri vnosu domenskega znanja v algoritem. Naloga zanke je poiskati kritične primere in jih ponuditi ekspertu za argumentiranje. Diagram poteka delovanja zanke je prikazan na sliki 3.1, opis korakov pa je opisan v nadaljevanju [17, 18]:

1. V prvem koraku algoritem ABML postavi hipotezo, ki se jo nauči na učnih podatkih. Naučeni model ali hipoteza je funkcija, ki zna preslikati nove vhodne podatke v izhodni razred oz. spremenljivko.
2. V drugem koraku algoritem poišče najbolj kritične primere. Kritični primeri so tisti primeri, ki jih postavljena hipoteza ne zna uvrstiti v pravilen razred. V primeru, da je kritičnih primerov več, se ti razvrstijo po stopnji kritičnosti. Najbolj kritičen primer postane tisti, ki ga

metoda napove z največjo napako.

3. Algoritem kritičen primer predstavi ekspertu. Če kritičnega primera ni, je postopek argumentiranja zaključen.
4. Ekspert razloži primer v naravnem jeziku. Če je primer nerazumljiv ali pa ga ne zna razložiti, lahko zahteva nov kritični primer.
5. Argumenti eksperta se preko aplikacije vnesejo v model.
6. Algoritem najde protiprimere, za katere vnešen pogoj velja, nahajajo pa se v nasprotnem razredu od argumentiranega primera.
7. Ekspertu je ponujena možnost, da argumente glede na protiprimere popravi ali dopolni.
8. Algoritem upošteva spremembe eksperta in ponovno izvede iskanje protiprimerov. V primeru, da protiprimerov ni, se algoritem nauči nove hipoteze in poišče najbolj kritičen primer.
9. Postopek se zaključi, ko ni več kritičnih primerov oz. kadar je ekspert zadovoljen z argumenti.



Slika 3.1: Diagram poteka argumentiranja.

Tabela 3.2: Primer učnih podatkov

ImeOsebe	Temperatura	Cepljen	Kašljanje	Glavobol	...	Gripa
g. Horvat	normalna	da	ne	ne	...	ne
ga. Novak	visoka	ne	da	ne	...	da
ga. Kranjc	zelo visoka	ne	ne	da	...	da
g. Kovačič	visoka	da	da	ne	...	ne
...

Prikaz argumentiranja si oglejmo na primeru za diagnosticiranje gripe.

Na voljo imamo podatke štirih oseb, prikazane v tabeli 3.2. Tem podatkom bomo rekli učni primeri. Vsak učni primer je opisan z različnimi atributi: *ImeOsebe*, *Temperatura*, *Cepljen*, *Kašljanje*, *Glavobol*, itd. in enim razredom *Gripa*. *ImeOsebe* je opisni atribut, ki ga bomo v nadaljevanju uporabili za lažjo razlago, ostali atributi (prikazani tudi v tabeli 3.3) pa se uporabljajo pri učenju. Atribut *Temperatura* je sicer številski atribut, ki smo ga zaradi bolj nazorne razlage spremenili v kategorijske vrednosti.

Predpostavimo, da se algoritem strojnega učenja nauči naslednje pravilo:
Če oseba nima zelo visoke temperature, potem nima gripe.

Iz primerov, ki jih imamo na voljo, opazimo, da zgornje pravilo ne pokrije vseh primerov. Pri ga. Novak velja, da nima zelo visoke temperature in hkrati ima gripo. Primer ga. Novak tako postane kritičen primer, ki ga je model klasificiral v napačen razred. Ekspert skuša z argumenti razložiti možen vzrok za napačno klasifikacijo. Poda naslednji argument:

Oseba ima gripo, če ima temperaturo večjo od normalne.

Algoritem strojnega učenja ekspertov argument uporabi in zgradi nov model. Ker se pojavi neskladje med argumentom eksperta in učnimi primeri, se pojavi protiprimer. Iz tabele opazimo, da ima oseba g. Kovačič visoko

temperaturo, vendar nima gripe. Ekspert s pomočjo primerjave kritičnega primera in protiprimera poišče razlog za odstopanje. V podatkih opazi, da ga. Novak ni bila cepljena proti gripi. Algoritem ekspertu omogoči dopolnitev argumenta:

Če ima oseba visoko temperaturo in ni bila cepljena proti gripi, potem ima gripo.

Z uporabo slednjega argumenta so pokriti vsi primeri. Iz zgornjega primera lahko opazimo, da je ena izmed ključnih prednosti zanke za zajem ekspertnega znanja samodejno iskanje protiprimerov, ki ekspertu omogočajo lažjo razlago kritičnega primera.

Ena od možnosti interaktivne zanke za zajem ekspertnega znanja je tudi vnos novih atributov, s čimer si v učno domeno vnese višjenivojske attribute, ki so skladni z njegovim razumevanjem domene. Predpostavimo, da imamo pri primeru za diagnosticiranje gripe na voljo dodatne attribute, ki so prikazani v tabeli 3.3.

Tabela 3.3: Primer podatkov z novim atributom

ImeOsebe	...	Glavobol	Izčrpanost	VnetoGrlo	Apetit	SimptomiGripe	Gripa
g. Horvat	...	ne	ne	da	normalen	ne	ne
ga. Novak	...	ne	da	da	nizek	da	da
ga. Kranjc	...	da	da	ne	nizek	da	da
g. Kovačič	...	ne	ne	ne	normalen	ne	ne
...

V tabelo 3.3 vpeljemo nov atribut *SimptomiGripe*, ki ga je ekspert sestavil iz obstoječih atributov: *Glavobol*, *Izčrpanost*, *VnetoGrlo* in *Apetit*. Nov atribut se doda med nabor podatkov. Z vpeljavo novega atributa lahko argumentiramo protiprimere in izpeljemo nov argument:

Če ima oseba simptome gripe in temperaturo večjo od normalne, potem ima gripo.

Zgornji argument pokrije vse dane primere.

Iskanje kritičnih primerov in protiprimerov lahko pozitivno vpliva na detekcijo napak v podatkih. V primeru, da se v podatkih pojavi nekonsistenca med pravili in klasificiranim razredom, to metoda odkrije in kot kritični primer prikaže napačno klasificiran primer. Primer je prikazan v tabeli 3.4.

Tabela 3.4: Kritični primer

ImeOsebe	Temperatura	Cepljen	...	VnetoGrlo	Apetit	SimptomiGripe	Gripa
...
ga. Kristan	normalna	da	...	ne	normalen	ne	da
...

Kot kritični primer se prikaže primer osebe ga. Kristan. Gospa nima simptomov gripe, visoke temperature in bila je cepljena. Iz podatkov sklepamo, da je prišlo do napake v podatkih in primeru klasifikacijo popravimo na *Gripa* = *ne* [19].

3.2 Pregled sorodnih del

Argumentirano strojno učenje je bilo prvič podrobneje predstavljeno v [16]. Avtorji so nazorno opisali implementacijo metode z argumentiranim strojnim učenjem ABCN2, ki je razširjena različica algoritma za učenje pravil CN2, in predstavili postopek za zajemanje ekspertnega znanja, ki pa v osnovni različici še ni vseboval protiprimerov. Le-ti so bili prvič predstavljeni pri uporabi argumentiranega strojnega učenja za namen prepoznavanja slabih lovcev v domeni šah [18]. S pomočjo postopka elicitacije znanja iz domenskih ekspertov in vpeljave višjenivojskih atributov se je napovedna točnost pri napovedovanju, ali je določen lovec na šahovnici dober ali slab, povečala iz začetnih 72 % na 95 % v končnem napovednem modelu [20].

V [18] avtorji navajajo znanje kot ključno komponento vsakega inteligentnega sistema. Za delovanje takega sistema je pomemben kakovosten zajem

znanja, kar je pogosto težavna naloga in predstavlja največjo oviro pri gradnji inteligentnih sistemov. S tem problemom so se soočali na več različnih načinov, kot so npr. intervjuji, opazovanje, analogija, vendar kljub temu to ostaja nerešljiv problem. Kot alternativo tem načinom so predlagali metodo argumentiranega strojnega učenja, ki omogoča interakcijo med metodo strojnega učenja in domenskim strokovnjakom. Ekspert na enostaven način, preko interaktivne zanke v algoritem strojnega učenja, vnaša svoje domensko znanje. Primer delovanja argumentiranega strojnega učenja ponazorijo na primeru slabega lovca (angl. *bad bishop*).

V [17] je podrobno opisana zanka za zajem ekspertnega znanja, ki smo jo opisali v poglavju 3.1 in uporabili pri izdelavi nove metode. Interaktivna zanka za zajemanje ekspertnega znanja ekspertovo pozornost usmeri v najbolj kritičen primer, kar olajša podajanje ustreznih argumentov.

V [21] avtorji članka predstavijo delovanje metode argumentiranega strojnega učenja na sistemu, ki bo nevrologom pomagal pri prepoznavi različnih vrst tresavic. Sistem služi kot drugo mnenje in je uporaben predvsem pri klasifikaciji težkih primerov, kar je privedlo do manjšega števila nadaljnjih preiskav. Končen rezultat je bil sistem, ki je skladen z domenskim znanjem in boljšo napovedno točnostjo.

V tabeli 3.5 so predstavljeni rezultati algoritmov pred in po vpeljavi metode za argumentirano strojno učenje. Iz tabele opazimo, da metoda ABML, ki omogoča zajem ekspertnega znanja, izboljša rezultate napovednih modelov.

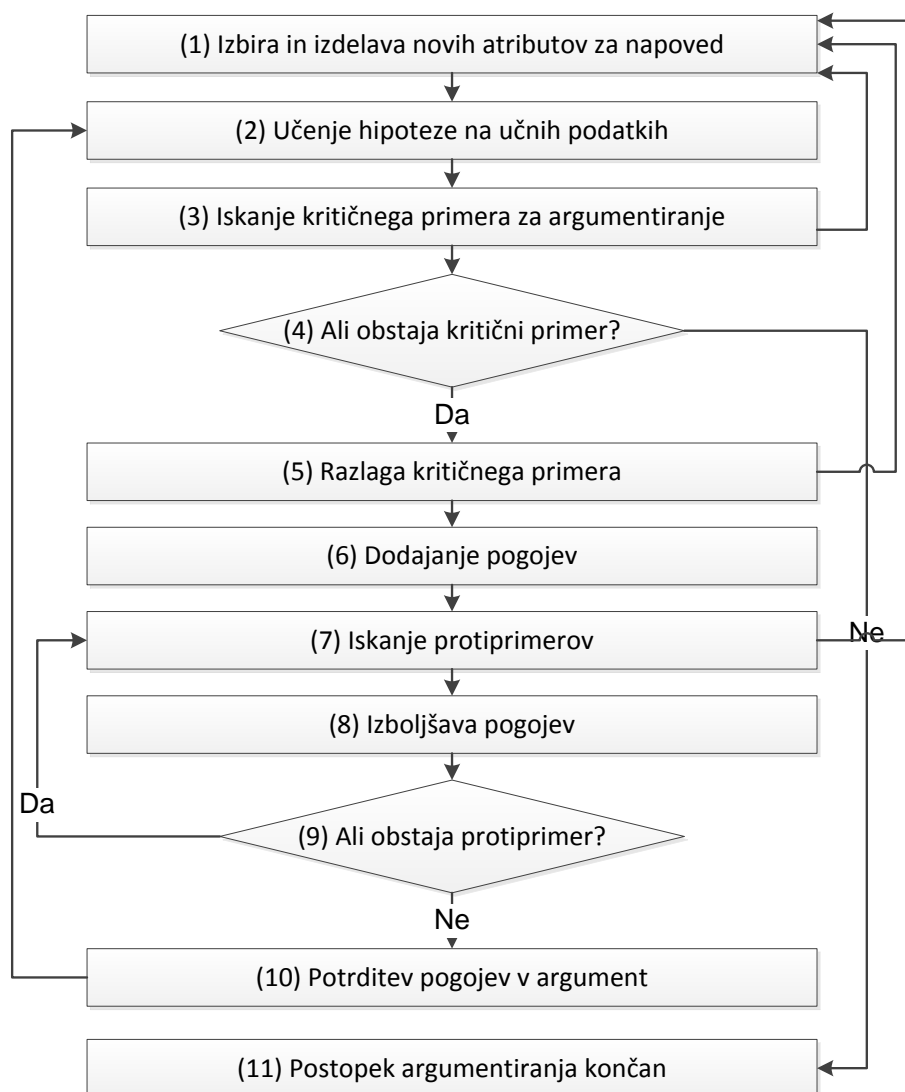
Tabela 3.5: Napovedne točnosti pred in po uporabi ABML

Domena	Problem	CA (pred)	CA (potem)	Referenca
živali	opisovanje vrste živali	94 %	97 %	Možina [5]
finance	bonitetne ocene	80 %	97 %	Pavlič, Možina et al. [7, 16]
šah	slabi lovec	72 %	95 %	Možina et al. [15, 18]
nevrologija	razlikovanje tresavic	82 %	91 %	Groznič et al. [21]
avtomobili	ocenjevanje kvalitete avtomobilov	91 %	95 %	Napierała et al. [22]

Poglavje 4

Logistična regresija z možnostjo argumentiranja

Metoda logistične regresije z možnostjo argumentiranja (angl. *argument-based logistic regression*) temelji na interaktivni zanki za zajemanje ekspertnega znanja (opisana v poglavju 3.1), ki temelji na strojnem učenju s pomočjo logistične regresije (opisana v poglavju 2.2). Preko novo izdelane metode v gradnjo napovednega modela vključimo dodatno znanje v obliki novih atributov. Za enostavno uporabo metode in lažjo interakcija med metodo strojnega učenja in ekspertom smo izdelali aplikacijo z grafičnim vmesnikom, ki ekspertu služi kot orodje za učenje in argumentiranje. Koraki nove metode so opisani v nadaljevanju, diagram poteka pa na sliki 4.1. Grafični vmesnik aplikacije je opisan v poglavju 5.



Slika 4.1: Diagram poteka nove metode.

4.1 Izbira in izdelava novih atributov za napoved

Prvi korak metode omogoča izbiro in izdelavo novih atributov, ki bodo uporabljeni pri postavitvi hipoteze. Med celotnim potekom argumentiranja primerov je uporabniku omogočeno spreminjanje nabora atributov, ki jih uporablja pri gradnji napovednega modela. Z možnostjo izbire atributov ekspertu zagotovimo boljši pregled nad pridobljenim modelom, hkrati pa mu omogočimo, da izbere samo tiste attribute, ki jih pozna ali pa od njih pričakuje, da bodo koristili pri boljši napovedni točnosti modela.

Za lažji izbor atributov se izračunajo njihove ocene s pomočjo večjega števila klasifikatorjev naključnih odločitvenih dreves (angl. *extra-trees classifier*). Pomembnost atributa izračunajo glede na globino atributa v posameznem odločitvenem vozlišču in glede na zmožnost posameznega atributa, da pravilno napove ciljni razred. Atributi, ki se nahajajo višje v drevesu, prispevajo večji delež h končni napovedi ciljnega razreda. Ker se zgradi večje število dreves, se rezultati na koncu povprečijo. S tem se zmanjša varianca in pristranskost med posameznimi drevesi. Metoda vrne seznam pozitivnih vrednosti, ki se seštevajo v 1. Bolj, kot je pomemben atribut za končno napoved, večja je njegova vrednost.

Ocene atributov ekspertu nudijo hiter vpogled v pomembnost posameznega atributa in olajšajo njihov izbor. Atributi so urejeni po oceni v padajočem vrstnem redu. Nove attribute lahko tvorimo z uporabo obstoječih atributov iz začetne množice podatkov ali pa uporabimo katerega od novo ustvarjenih atributov. Med poljubno izbranimi atributoma izberemo eno izmed naslednjih aritmetičnih operacij: *seštevanje*, *odštevanje*, *množenje*, *deljenje*. Attribute, ki jih vsebuje napovedni model, lahko poljubno dodajamo in odstranjujemo med samim argumentiranjem oz. učenjem. Pri tem je potrebno omeniti, da se ob vsaki spremembi izbora atributov napovedni model ponovno izgradi. Slednje lahko vodi tudi do zamenjave kritičnega primera.

4.2 Učenje hipoteze na učnih podatkih

V drugem koraku se algoritem nauči hipoteze nad izbranimi atributi in učno množico podatkov. Vsi podatki se pred izdelavo modela pretvorijo v številske vrednosti in standardizirajo. Model s pomočjo naučene hipoteze napove izhodne vrednosti razredov na testni množici podatkov.

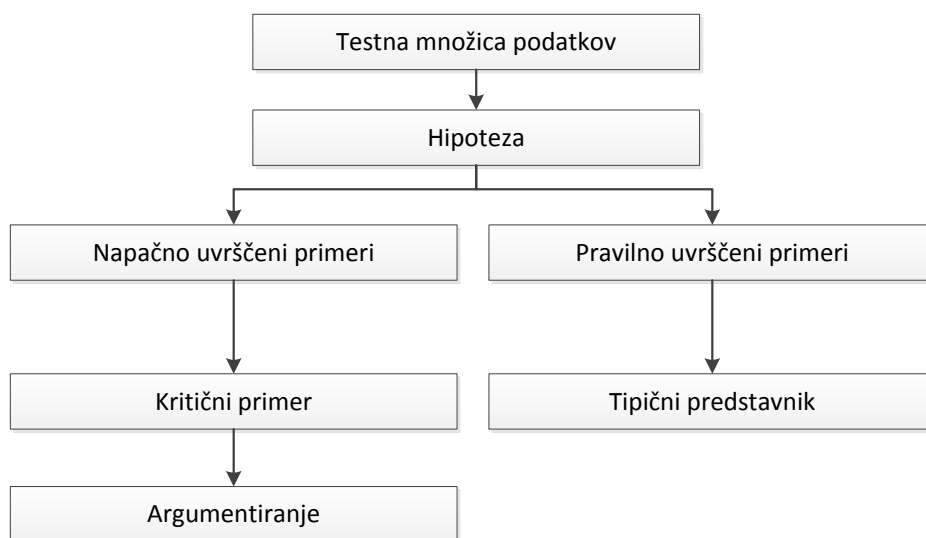
4.3 Iskanje kritičnega primera

Tretji korak poišče najbolj problematičen primer, ki ga je model klasificiral v napačen razred. Tisti primer, ki ga metoda z največjo verjetnostjo uvrsti v napačen razred, imenujemo *kritični primer*. Metoda napačno uvrščene primere razvrsti po verjetnosti od tistega z največjo verjetnostjo napačnega razreda do tistega z najmanjšo. Izdelana je tako, da v primeru, da uporabnik spremeni nabor atributov, pride do spremembe modela, kar povzroči ponovno gradnjo modela in ponoven izračun kritičnih primerov.

4.4 Razlaga kritičnega primera

V četrtem koraku metoda vse primere razvrsti v dva seznama, urejena po verjetnosti: seznam *pravilno uvrščenih* primerov in seznam *napačno uvrščenih* primerov. V seznamu napačno uvrščenih primerov najdemo primere, ki jih metoda ne zna uvrstiti v pravilen razred. Najbolj kritičen primer iz seznama napačno uvrščenih primerov je predstavljen ekspertu. Ker obstaja možnost, da ekspert določenega primera ne zna pojasniti, mu je omogočena zamenjava kritičnega primera. Če se ekspert odloči zamenjati kritični primer, to postane naslednji primer, ki ga je metoda z največjo verjetnostjo uvrstila v napačen razred. V primeru, da kritičnega primera ni, je postopek argumentiranja zaključen.

Za lažje argumentiranje kritičnega primera metoda poišče tipičnega predstavnika nasprotnega razreda. Tipičen predstavnik nasprotnega razreda se



Slika 4.2: Iskanje kritičnega primera in tipičnega predstavnika.

nahaja v seznamu pravilno uvrščenih primerov. To postane primer z največjo verjetnostjo, ki je klasificiran v pravilen razred in se nahaja v razredu, kamor bi se moral uvrstiti kritični primer. Postopek je prikazan na diagramu poteka na sliki 4.2

Za pomoč pri razlagi kritičnega primera je ekspertu na voljo grafični prikaz kvantilov (angl. *box plot*), ki prikazuje statistiko nad podatki izbranega atributa, in prej omenjeni tipični predstavnik nasprotnega razreda.

4.5 Argumentiranje

Argument je lahko sestavljen iz več pogojev. Glede na vnešen pogoj eksperta se sproti prikazujejo protiprimeri. Pogoj je določen z:

1. izbiro poljubnega atributa iz nabora izbranih atributov,
2. izbiro enega izmed primerjalnih operatorjev (glej tabelo 4.1) in

3. vnosa poljubne vrednosti (opciski – v primeru, da uporabnik ne vpiše vrednosti, se upošteva vrednost atributa kritičnega primera).

Glede na vnešeno vrednost eksperta metoda poišče protiprimere. Ekspert vedno vnaša pogoje, ki predstavljajo razlog, da kritičen primer sodi v pravi razred, se pravi v nasproten razred, kot ga je klasificiral model logistične regresije.

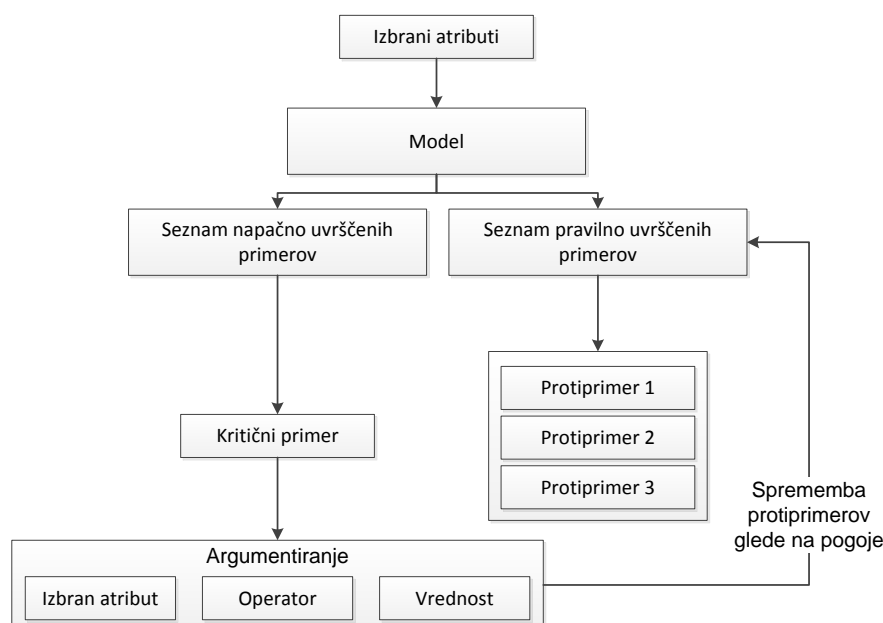
Tabela 4.1: Primerjalni operatorji in njihov opis

Operator	Opis
\geq	Kritičen primer sodi v nasproten razred, ker je vnešena vrednost večja ali enaka vrednosti atributa kritičnega primera.
\leq	Kritičen primer sodi v nasproten razred, ker je vnešena vrednost manjša ali enaka vrednosti atributa kritičnega primera.
$=$	Kritičen primer sodi v nasproten razred, ker je vnešena vrednost enaka vrednosti atributa kritičnega primera.
\neq	Kritičen primer sodi v nasproten razred, ker vrednost atributa kritičnega primera ni enaka vnešeni vrednosti.

4.6 Iskanje protiprimerov

Protiprimeri se generirajo glede na vnosno vrednost pogoja. Vsi prikazani protiprimeri zadovoljujejo podanemu pogoju, vendar s pomembno razliko, da pripadajo nasprotnemu razredu. Logika za iskanje protiprimerov je narejena tako, da v testni množici podatkov poišče pravilno uvrščene primere nasprotnega razreda in glede na podan atribut, operator in vrednost v pogoju eksperta poišče tiste primere, ki ustrezajo pogoju. Prikazujemo največ tri protiprimere, ki imajo največjo verjetnost nasprotnega razreda. Prikaz iskanja protiprimerov je prikazan na sliki 4.3.

Ker se ekspert lahko v določenem trenutku zmoti pri definiciji pogoja ali



Slika 4.3: Diagram poteka iskanja protiprimerov.

s pomočjo protiprimerov ugotovi, da bi bilo posamezen pogoj mogoče izboljšati, lahko katerikoli pogoj kadarkoli izbriše in doda novega.

V koraku devet se preveri obstoj protiprimerov glede na podane pogoje. V primeru, da še vedno obstajajo protiprimeri, so le-ti prikazani ekspertu. Če ni najdenega protiprimera, se postopek nadaljuje s potrditvijo danih pogojev v argument. Ekspert je lahko v kateremkoli trenutku zadovoljen s podanimi pogoji in pogoje potrdi v argument. Z izdelavo novega argumenta se ponovno izračuna hipoteza, ki poišče nov protiprimer.

4.7 Pridobivanje znanja iz argumentov

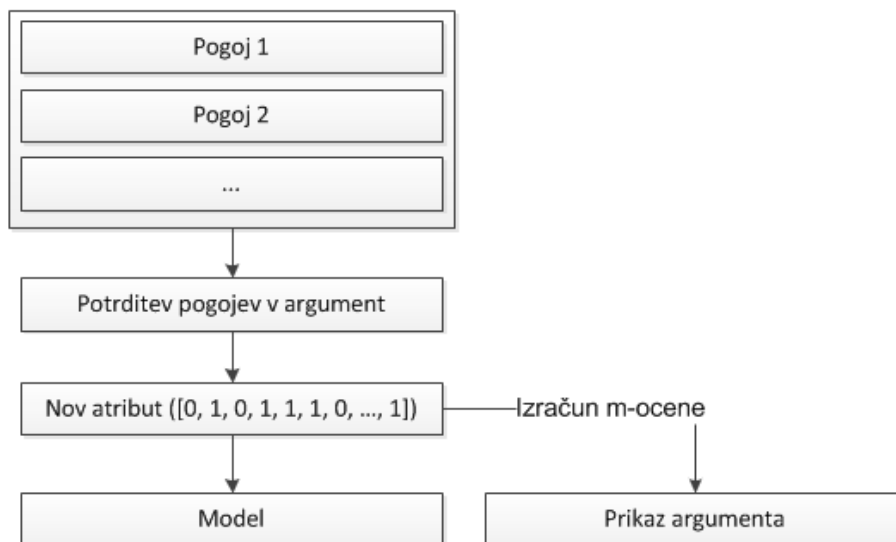
Argument, ki je sestavljen iz enega ali več pogojev, se kot dodatni atribut doda v množico podatkov. Nov atribut ima vrednost 1 pri vsakem učnem primeru, ki zadovoljuje vsem podanim pogojem v argumentu in vrednost 0

za primere, kjer pogoji niso zadoščeni. Ker ločimo pozitivne in negativne argumente glede na razred, ki ga argumentiramo, moramo to upoštevati pri gradnji modela. Postavimo omejitev, da je parameter modela θ_i lahko le pozitivno število pri argumentih, ki napovedujejo vrednost razreda $y = 1$, in negativno število pri argumentih, ki napovedujejo vrednost razreda $y = 0$.

Delovanje navedenega mehanizma pojasnimo na primeru. Predpostavljamo, da imamo nabor atributov $[x_0, x_1, x_2, \dots, x_n]$ in pripadajoče parametre modela $[\theta_0, \theta_1, \theta_2, \dots, \theta_n]$. Naš novi atribut označimo z x_i in njegov pripadajoči parameter modela θ_i . V primeru, da je argument naklonjen razredu $y = 1$, bi želeli z argumentom in parametrom modela povečati verjetnost za pozitiven razred. Iz formule (4.1) opazimo, da se vrednosti uteženo seštejejo, kar je razlog, da bi imeli vrednost θ_i čim večje pozitivno število. Tako bo argument pozitivno vplival na napoved razreda $y = 1$. Pri argumentu, ki je naklonjen razredu $y = 0$, pa je logika obratna. Tu bi želeli imeti za parameter modela negativno število, saj želimo zmanjšati verjetnost za razred $y = 1$ in se približati negativnemu razredu.

$$\theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n + \theta_i x_i \quad (4.1)$$

Ena od pomembnih lastnosti je tudi prikaz argumenta. Ker je argument v modelu predstavljen kot dodaten atribut, tega nikoli ne prikažemo ekspertu. Pomembno je, da se argument ne prikaže kot možnost pri izboru atributov ali pri prikazu kritičnega primera in njegovih protiprimerov. Ekspert argument vedno vidi le kot pravilo za določanje razreda. Prikaz poteka izdelave argumenta je prikazan na sliki 4.4.



Slika 4.4: Izdelava novega argumenta.

Za vsak izdelan argument se izračuna tudi njegova ocena. Ocena argumenta ekspertu služi kot povratna informacija o kakovosti argumenta pri določanju ciljnega razreda. Oceno argumenta lahko izmerimo na npr. naslednja dva načina:

- m-ocena,
- napovedna točnost modela.

Za ocenjevanje kakovosti podanih argumentov eksperta uporabljamo m-oceno, ki nam takoj poda povratno informacijo o kvaliteti argumenta [23]. Pri tem je potrebno opomniti, da m-ocena služi le kot povratna informacija o kvaliteti argumenta, ki pa sama po sebi ne zagotavlja, da bo vsak dober ali slab argument izboljšal oz. poslabšal napovedno točnost modela. Izračun m-ocene je podan s formulo (4.2):

$$\mathbf{p} = \frac{r + mPa}{n + m} \quad (4.2)$$

- \mathbf{r} – število pozitivnih pokritih primerov argumenta

- **n** – število vseh primerov
- **Pa** – apriorna verjetnost ali pričakovanje, da je primer pozitiven
- **m** – parameter metode

Druga mera, ki bi jo lahko uporabili za oceno posameznega argumenta je sama sprememba napovedne točnosti algoritma. V primeru, da se napovedna točnost izboljša, bi lahko rekli, da nov argument pozitivno vpliva na domeno podatkov.

Metoda omogoča izbris poljubnega argumenta. Izbrisan argument se izbriše tudi iz modela. Ker gre za spremembo modela, se parametri modela ponovno izračunajo. Z možnostjo izbrisa argumentov uporabniku omogočimo popravljanje napak.

Postopek se zaključi, ko ni na voljo več nobenega kritičnega primera, kadar je ekspert zadovoljen z argumenti ali napovedno točnostjo modela oz. katero drugo mero. Pomembno je, da ima ekspert možnost zaključiti argumentiranje v vsakem trenutku, saj se tako izognemo prevelikemu prileganju podatkov (angl. *overfitting*).

Poglavje 5

Grafični vmesnik

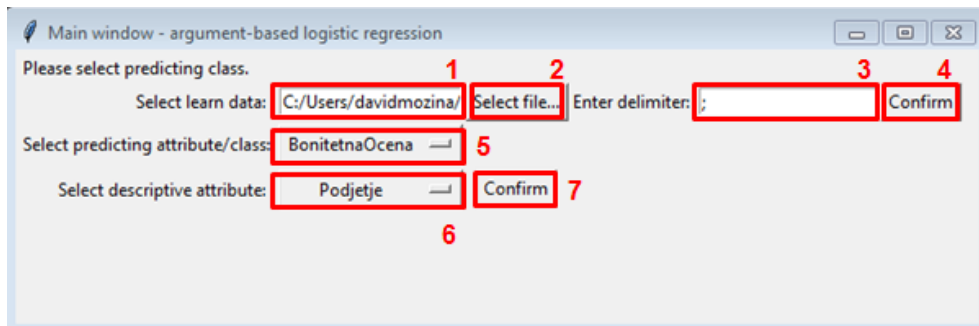
Aplikacija z grafičnim vmesnikom je bila narejena v sklopu implementacije metode logistične regresije z možnostjo argumentiranja. Ekspertu omogoča lažjo predstavitev podatkov, enostavno spoznavanje z novo vsebino oz. novo domeno podatkov in vključuje novo izdelano metodo, ki omogoča interakcijo med domenskim ekspertom in metodo logistične regresije. Aplikacija je izdelana s programskim jezikom Python in knjižnico Tkinter. Slednji je najbolj uporabljan paket za izdelavo grafičnih vmesnikov (angl. *GUI* oz. *graphical user interface*) [24]. Aplikacija je prosto dostopna na spletu [25].

5.1 Okno za izbor podatkov

Aplikacija je izdelana tako, da omogoča poljuben tekstovni vir podatkov. Pomembni sta le dve lastnosti vhodnih datotek: besedilo v datoteki mora biti v narekovajih in vse decimalne številke morajo imeti decimalno piko.

Za vnos množice podatkov je v aplikaciji pripravljen obrazec (glej sliko 5.1), ki uporabniku omogoča izbiro datoteke in izbor napovednega razreda.

S klikom na gumb *Select file...* (2) se odpre pogovorno okno, kjer je možen izbor datoteke na računalniku. Pot do datoteke se vpiše v vnosno polje (1). V vnosno polje (3) je potrebno vnesti ločilo med podatki in izbor potrditi s klikom na gumb *Confirm* (4). Prikažeta se dva spustna seznama. Prvi



Slika 5.1: Okno za izbor podatkov.

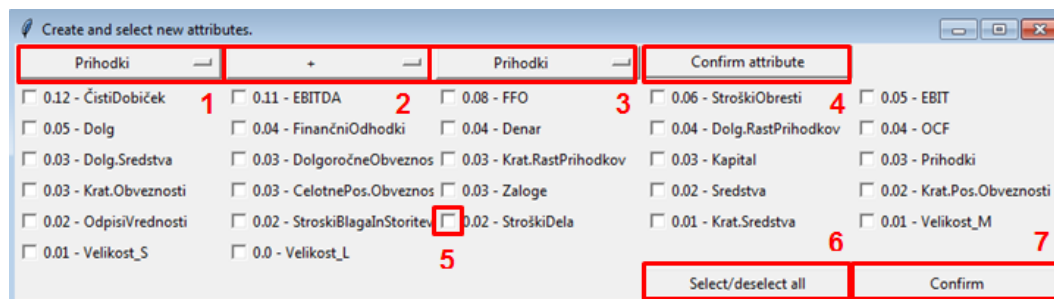
spustni seznam je namenjen izbiri napovednega atributa (5). Napovedni atribut mora imeti eno izmed naslednjih vrednosti:

- 1 ali 0,
- "da" ali "ne",
- "yes" ali "no",
- "true" ali "false".

Spustni seznam za izbiro opisnega atributa (6) pa je poljuben. Privzeto je izbrana možnost "None". V primeru izbire katerega od opisnih atributov se ta atribut odstrani iz množice podatkov. Prikaže se pri kritičnem primeru in njegovih protiprimerih za lažje argumentiranje. Prav tako ne vpliva na napovedno točnost modela. Izbrane možnosti potrdimo s klikom na gumb *Confirm* (7). Odpre se okno, ki omogoča izbiro atributov.

5.2 Okno za izbor atributov

V oknu za izbiro in izdelavo novih atributov (glej sliko 5.2) ob izbiri poljubnega atributa iz prvega (1) in drugega (3) spustnega seznama (angl. *drop-down list*) in želene operacije med atributoma (2) izdelamo nov atribut s potrditvijo na gumb *Confirm attribute* (4). S klikom na gumb za potrditev



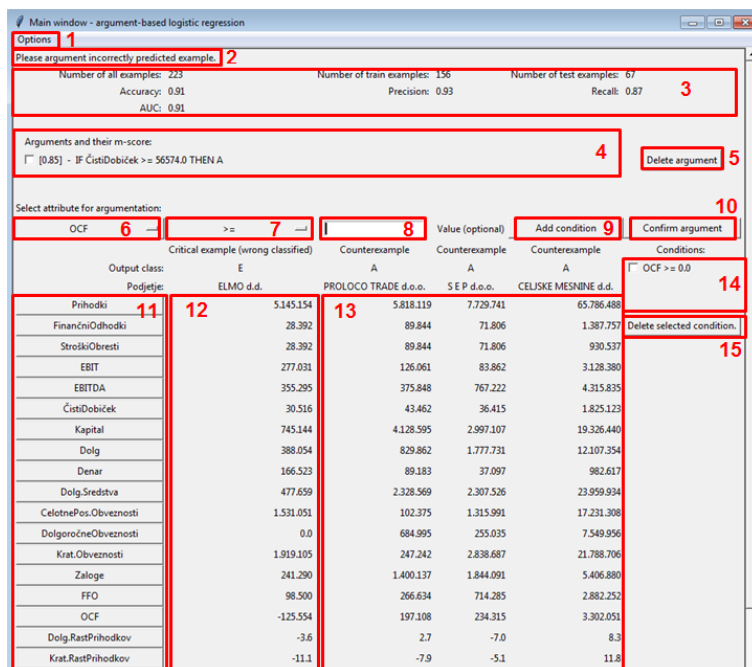
Slika 5.2: Okno za izbor atributov.

izdelave atributa se ponovno izračunajo vrednosti za pomembnost atributa. Atributi se uredijo po padajočem vrstnem redu glede na oceno atributa. Z izbiro v poljubnem stikalu (angl. *checkbox*) (5) označimo, katere attribute bi želeli imeti v modelu. Za hitrejšo izbiro atributov se na okencu nahaja gumb *Select/deselect all* (6), ki omogoča označitev ali odznačitev vseh atributov. Ko imamo vse želene attribute izbrane, izbiro potrdimo s klikom na gumb *Confirm* (7). S potrditvijo se v ozadju zgradi model, ki vsebuje izbrane attribute.

5.3 Okno za argumentiranje primerov

Glavno okno v aplikaciji je namenjeno spoznavanju podatkov, detekciji možnih napak pri klasifikaciji razreda in izdelavi modela, ki je skladen z ekspertnim znanjem. Izgled glavnega okna je prikazan na sliki 5.3.

V orodni vrstici se nahaja izbira *Options* (1), ki omogoča ponovni prikaz okna za izbiro atributov in izhod iz aplikacije. Pod orodno vrstico se nahaja statusna vrstica (2), ki uporabnika z navodili vodi do pravilne rabe aplikacije, in informacijsko polje (3), ki prikazuje vrednosti mer za ocenjevanje napovednega modela. Mere, ki jih aplikacija prikazuje, so napovedna točnost (angl. *accuracy score*), preciznost (angl. *precision*), priklic (angl. *recall*) in AUC (angl. *area under curve*), to je površina pod krivuljo ROC (angl. *receiver operating characteristic*). Ob vsaki spremembi modela (npr. dodanem no-



Slika 5.3: Glavno okno za argumentiranje.

vem atributu) ali pri vnosu novega argumenta se vrednosti omenjenih ocen ponovno izračunajo.

Pod informacijskim poljem se nahaja polje za prikaz zgodovine argumentov (4). Polje vsebuje vse argumente, ki jih uporabnik vnese v model. Funkcionalnost prikaza zgodovine argumentov skozi celotno učenje uporabniku omogoča vpogled v pridobljeno znanje. Z izbiro stikala pred posameznim argumentom in potrditvijo na gumb *Delete argument* (6) se argument odstrani iz polja in napovednega modela.

Sledijo glavne komponente metode za argumentiranje. Spustni seznam (6) je namenjen izbiri atributa, ki ga želimo argumentirati. Na voljo so vsi atributi, ki smo jih izbrali v oknu za izbiro atributov (11). V polje za vnos besedila (8) se vpiše poljubno število, ki bo glede na izbiro iz spustnega seznama za izbor matematičnega simbola (7) sestavljalo pogoj. Vnos števila v polje za vnos vrednosti je opsijski. Za pomoč uporabniku je s klikom na posamezni atribut (11) omogočen ogled statistike posameznega atributa. V

primeru, da polje ostane prazno, se vzame vrednost, ki jo vsebuje kritični primer (12) za argumentiran atribut (6). Pogoje potrdimo s klikom na gumb *Add condition* (9). Pogoje se doda v polje za prikaz trenutnih pogojev (14). Ker se ekspert pri navajanju pogojev lahko tudi zmoti, je dodana možnost za izbris posameznega pogoja. Prikaz protiprimerov (13) ekspertu omogoča lažje argumentiranje. Za pogoj, ki ga želimo izbrisati, označimo stikalo pred vrednostjo pogoja in izbris pogoja potrdimo z gumbom *Delete selected condition*. V trenutku, ko smo zadovoljni s pogojem in bi radi končali argumentiranje kritičnega primera, s klikom na gumb *Confirm argument* (10) pogoje spremenimo v argument. Model se ponovno preračuna in prikaže najbolj kritičen primer, argument pa se doda v polje za argumente (4).

Poglavje 6

Evalvacija

Uspešnost metode logistične regresije z možnostjo argumentiranja smo evalvirali in preverili na naslednja načina:

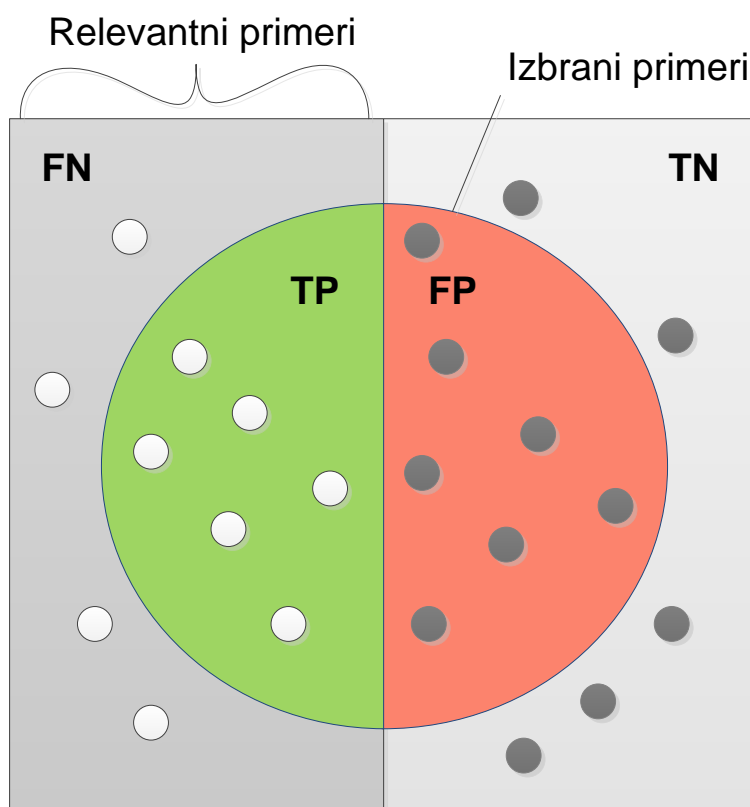
- spremembe ocen napovednega modela z izdelavo novih atributov,
- z ekspertom, ki smo ga prosili za argumentiranje kritičnih primerov.

6.1 Mere za evalvacijo

Delovanje algoritma bomo evalvirali z naslednjimi merami: napovedna točnost, preciznost, priklic in AUC. Z njihovo pomočjo dobimo vpogled v gibanje napovedne točnosti, napovedno moč atributov in argumentov. Za lažjo predstavo definirajmo nekaj pojmov:

- TP – Število pravilno klasificiranih pozitivnih primerov (dejanski razred je $y = 1$, napovedan razred je $y = 1$).
- FP – Število primerov, ki smo jih klasificirali kot pozitivne, v resnici pa so negativni.
- FN – Število primerov, ki smo jih klasificirali kot negativne, v resnici pa so pozitivni.

- TN – Število pravilno klasificiranih negativnih primerov (dejanski razred je $y = 0$, napovedan razred je $y = 0$).



Slika 6.1: Evalvacija uvrščenih primerov.

Napovedna točnost:

Funkcija za izračun napovedne točnosti (6.1) izračuna število pravilno klasificiranih primerov. Vrne odstotek pravilnih napovedi, kjer je napovedan razred \hat{y} enak dejanskemu izhodnemu razredu y .

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (6.1)$$

Preciznost:

Preciznost interpretiramo kot razmerje med resničnimi pozitivno klasificiranimi primeri in med primeri, ki jih pozitivno napovemo. Definiramo ga z enačbo (6.2).

$$\text{Precision} = \frac{TP}{TP + FP} \quad (6.2)$$

Priklic:

Priklic (6.3) interpretiramo kot razmerje med resničnimi pozitivno klasificiranimi primeri in med primeri, ki so resnično pozitivni.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (6.3)$$

Površina pod krivuljo ROC:

Površino pod krivuljo ROC (AUC) interpretiramo kot verjetnost, da bo klasifikator z naključno izbiro pozitivnega in negativnega primera višje uvrstil pozitivni primer. Večja, kot je vrednost površine pod krivuljo ROC, boljši je model.

6.2 Vpliv novih atributov na napovedno točnost

Aplikacija omogoča izdelavo novih atributov, ki so ekspertu v pomoč pri argumentiranju. Začetnemu modelu bomo dodali nove attribute in primerjali rezultate glede na začetno stanje modela. Na začetku bomo imeli izbrane vse attribute iz nabora podatkov. Izdelali bomo nekaj novih atributov in jih enega za drugim dodajali napovednemu modelu ter merili spremembo napovedne točnosti glede na začetno stanje modela. Vpliv atributov bomo izmerili na naslednjih algoritmihi: logistična regresija, naključna drevesa, metoda podpornih vektorjev in logistična regresija z možnostjo argumentiranja.

6.2.1 Opis podatkov

Za eksperiment bomo vzeli podatke o sladkornih boleznih indijanskega plemena Pima (angl. *Pima Indians Diabetes Data Set*), ki ga najdemo v shrambi podatkov UCI (angl. *UCI machine learning repository*) [26]. Množica podatkov je sestavljena iz 768 primerov, ki vsebujejo osem opisnih atributov in en binarni oz. napovedni atribut. Atributi so predstavljeni v tabeli 6.1.

Tabela 6.1: Opis atributov

Ime atributa	Opis
pred	število zanositev
plas	koncentracija glukoze v plazmi (2–urni test)
pres	pritisk v arterijah
skin	debelina kožne gube nadlahti
test	test insulina v krvi (2–urni test)
mass	indeks telesne mase
pedi	vpliv rodovnika na sladkorno bolezen
age	starost pacienta
class	ali ima pacient sladkorno bolezen

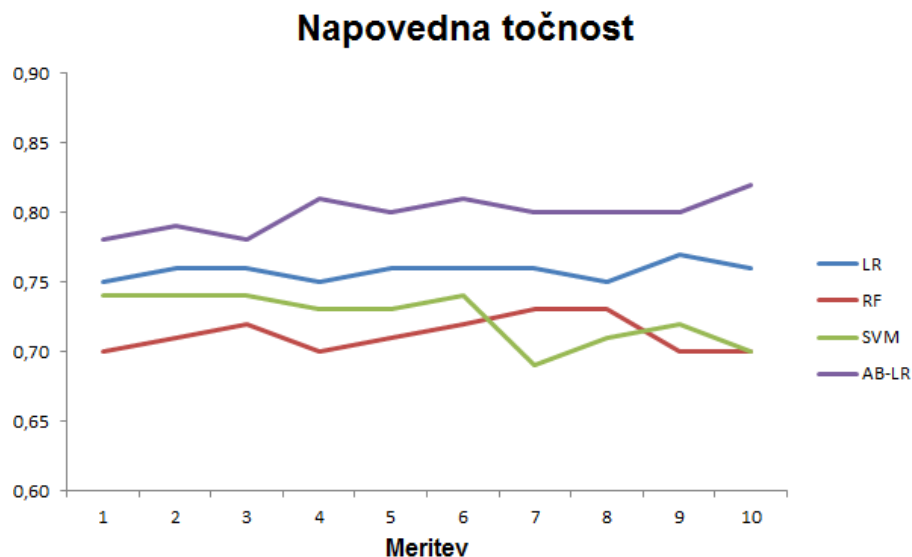
6.2.2 Eksperiment

V eksperimentu bomo skušali prikazati, kako novo izdelani atributi vplivajo na napovedno točnost modela. Dodali bomo nove attribute, ki jih bomo zgradili z uporabo osnovnih matematičnih operacij. Potek dodajanja atributov je opisan v stolpcu *Komentar* v tabeli 6.2, meritve pa so prikazane na slikah 6.2, 6.3, 6.4 in 6.5.

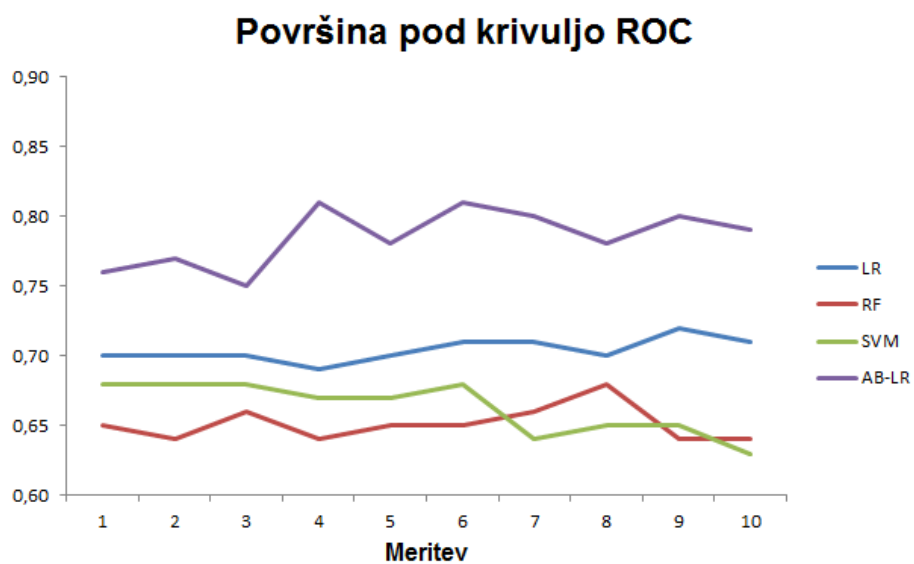
Iz podanih meritev opazimo, da izdelava novih atributov pripomore k boljšim napovednim modelom. Attribute smo dodajali povsem naključno in dokazali, da novi atributi ne le izboljšajo napovedno točnost logistične re-

Tabela 6.2: Potek izdelave novih atributov

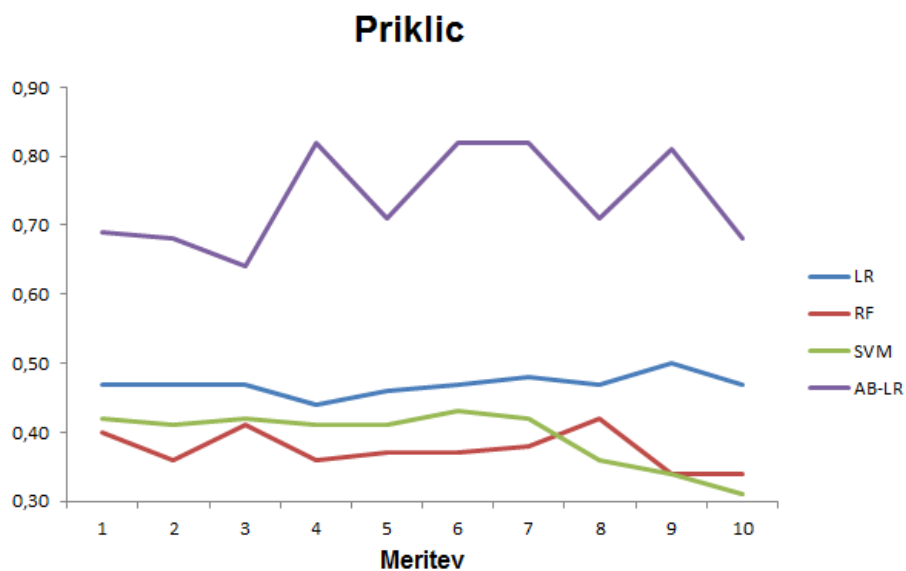
Iteracija	Komentar
1	Začetni atributi.
2	Dodamo $plas*plas$ zaradi najboljše ocene atributa.
3	Dodamo $mas*mas$, ker ima drugo najboljšo ocene atributa).
4	Vse ostale attribute kvadriramo.
5	Dodamo atribut $age*mass$.
6	Odstranimo $skin$ in $skin*skin$ atributa, zaradi slabe ocene.
7	Vse attribute kubiramo in označim vse.
8	Izberemo najboljših 10 atributov.
9	Naključno dodamo nove attribute ($*$ in \div). Izberemo najboljših 15.
10	Izberemo najboljših 20.



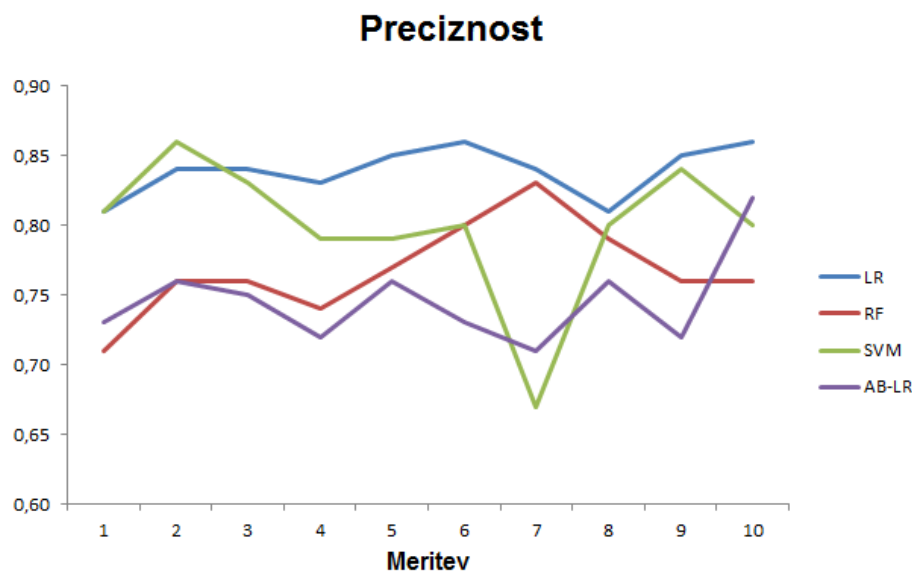
Slika 6.2: Gibanje napovedne točnosti različnih modelov.



Slika 6.3: Gibanje površine pod krivuljo ROC različnih modelov.



Slika 6.4: Gibanje priklica različnih modelov.



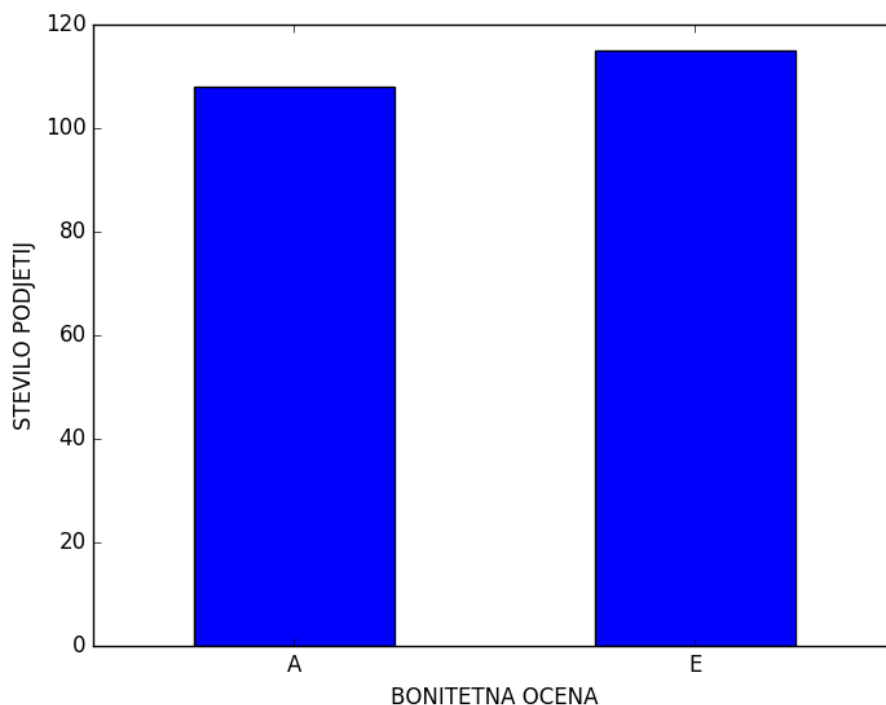
Slika 6.5: Gibanje preciznosti različnih modelov.

gresije, pač pa tudi drugih metod strojnega učenja, ki smo jih uporabili v eksperimentu.

6.3 Zajemanje znanja iz eksperta

6.3.1 Priprava podatkov

Delovanje metode logistične regresije smo z ekspertom testirali na bančnih podatkih, ki vsebujejo finančne izkaze za izračun bonitetne ocene podjetij. Finančni izkazi so vsebovali podatke za leta 2010, 2011, 2012 in 2013. Podatki za finančne ocene so pridobljeni iz aplikacije Gvin [27] in prikazujejo poslovanje podjetja v preteklem letu. Ker so možne bonitetne ocene označene s črkami od A do E, smo te, za delovanje logistične regresije z dvema atributoma, spremenili v dva razreda, prikazano na sliki 6.6. V razred A so bila uvrščena podjetja z bonitetno oceno A in B, v razred E pa podjetja z bonitetno oceno C, D in E. Pri opisu atributov smo si pomagali z viroma [28] in [29].



Slika 6.6: Distribucija podjetij po bonitetnih ocenah.

6.3.2 Opis podatkov

Podatki so opisani z naslednjimi atributi:

1. VelikostPodjetja

Opisuje velikost podjetja. Ima naslednji nabor vrednosti:

- **S** (angl. *small*) – majhno podjetje
- **M** (angl. *medium*) – srednje podjetje
- **L** (angl. *large*) – veliko podjetje

2. Prihodki

Prihodki od prodaje. Količina prodanega blaga in/ali storitev.

3. StroškiBlagaInStoritev

Stroški nabavne vrednosti prodanega blaga, materiala in storitev.

4. Stroški Dela

Stroški dela. Zajemajo plače zaposlenih, nadomestila in dajatve.

5. Odpisi Vrednosti

Odpisi vrednosti, kot so npr. stroški amortizacije in prevrednotenje virov sredstev.

6. Finančni Odhodki

Finančni odhodki so odhodki za financiranje in za naložbenje.

7. Stroški Obresti

Del finančnih stroškov, ki predstavlja obresti.

8. EBIT

Poslovni izid iz poslovanja pred obrestmi in davki.

9. EBITDA

Poslovni izid pred obrestmi, davki in amortizacijo.

10. Čisti Dobiček

Razlika med vsemi prihodki in odhodki, zmanjšana za davek.

11. Sredstva

Vsa sredstva podjetja.

12. Kapital

Kapital podjetja (vplačani kapital, kapitalske rezerve, ...). Izraža obveznost podjetja do lastnikov.

13. Dolg

Finančna zadolženost podjetja.

14. Denar

Denarna sredstva na računih in gotovina.

15. Dolg.Sredstva

Dolgoročna sredstva so sredstva, ki jih ima gospodarska družba dolgoročno za proizvodnjo proizvodov oz. opravljanje storitev (neopredmetena: dobro ime, patenti, stroški razvoja, licence, blagovne znamke, in opredmetena sredstva: zemljišča, zgradbe, oprema).

16. Kratk.Sredstva

Kratkoročna sredstva so sredstva, ki se praviloma preoblikujejo v časovnem obdobju, krajšim od enega leta. Gre za sredstva za prodajo, zaloge, finančna sredstva, kratkoročne poslovne terjatve, kratkoročna dana posojila in depozite.

17. Celotne.Pos.Obv.

Celotne poslovne obveznosti podjetja oz. dolg povezan s poslovanjem.

18. Kratk.Pos.Obv.

Kratkoročne poslovne obveznosti (kratkoročne obveznosti do dobaviteljev, zaposlenih, za davke). Zapadejo v roku enega leta.

19. Dolg.Obv.

Dolgoročne obveznosti so obveznosti z rokom zapadlosti plačila, daljšim od leta dni od njihovega nastanka (dolgoročne poslovne obveznosti, finančne obveznosti, ...).

20. Kratk.Obv.

Kratkoročne obveznosti so obveznosti v zvezi s financiranjem lastnih sredstev, ki zapadejo v plačilo v obdobju, krajšem od leta dni (finančne obveznosti, poslovne obveznosti, kratkoročna prejeta posojila, ...).

21. Zaloge

Zaloge surovin, materiala, blaga, nedokončanih proizvodov in gotovih proizvodov, trgovskega blaga.

22. FFO

Sredstva iz poslovanja.

23. OCF

Denarni tok iz poslovanja. Mera za določanje količine denarja iz običajnega poslovanja podjetja.

24. Dolg.RastPrihodkov

Dolgoročna rast prihodkov v obdobju daljšim od enega leta.

25. Krat.RastPrihodkov

Kratkoročna rast prihodkov v zadnjem letu.

26. BonitetnaOcena

Bonitetna vrednost podjetja. Izhodni razred, ki ga napovedujemo.

27. Dejavnost

Poslovna dejavnost, s katero se podjetje ukvarja.

6.3.3 Novi atributi

Aplikacija omogoča izdelavo novih atributov, ki bodo ekspertu v pomoč pri argumentiranju. Pred začetkom argumentiranja bomo izdelali naslednje attribute:

1. CurrentRatio

Kazalnik likvidnosti (6.4) meri sposobnost podjetja za plačilo kratkoročnih in dolgoročnih obveznosti.

$$\frac{Krat.Sredstva}{Krat.Obv.} \quad (6.4)$$

2. NetDebtToEBITDARatio

Sposobnost odplačila dolga na dolgi rok. Kazalnik (6.5) pove koliko let ob konstantni vrednosti EBITDA podjetje potrebuje za odplačilo dolga. V primeru, da ima podjetje več denarja kot dolga ali negativno vrednost EBITDA, bo kazalnik negativen.

$$\frac{Dolg - Denar}{EBITDA} \quad (6.5)$$

3. **EquityRatio**

Kazalnik (6.6) prikazuje delež kapitala v celotnih sredstvih in pove, koliko sredstev je financiranih z lastnim kapitalom.

$$\frac{Kapital}{Sredstva} \quad (6.6)$$

4. **TIE** (angl. *times interest earned*)

Kazalnik (6.7) meri sposobnost družbe, da s svojimi prihodki pokrije stroške obresti.

$$\frac{EBIT}{StroskiObresti} \quad (6.7)$$

5. **ROA** (angl. *return of assets*)

Donosnost sredstev (6.8) je kazalnik, ki pove, kako donosno je podjetje. Večja je vrednost kazalnika, uspešnejše je poslovanje.

$$\frac{EBIT}{Sredstva} \quad (6.8)$$

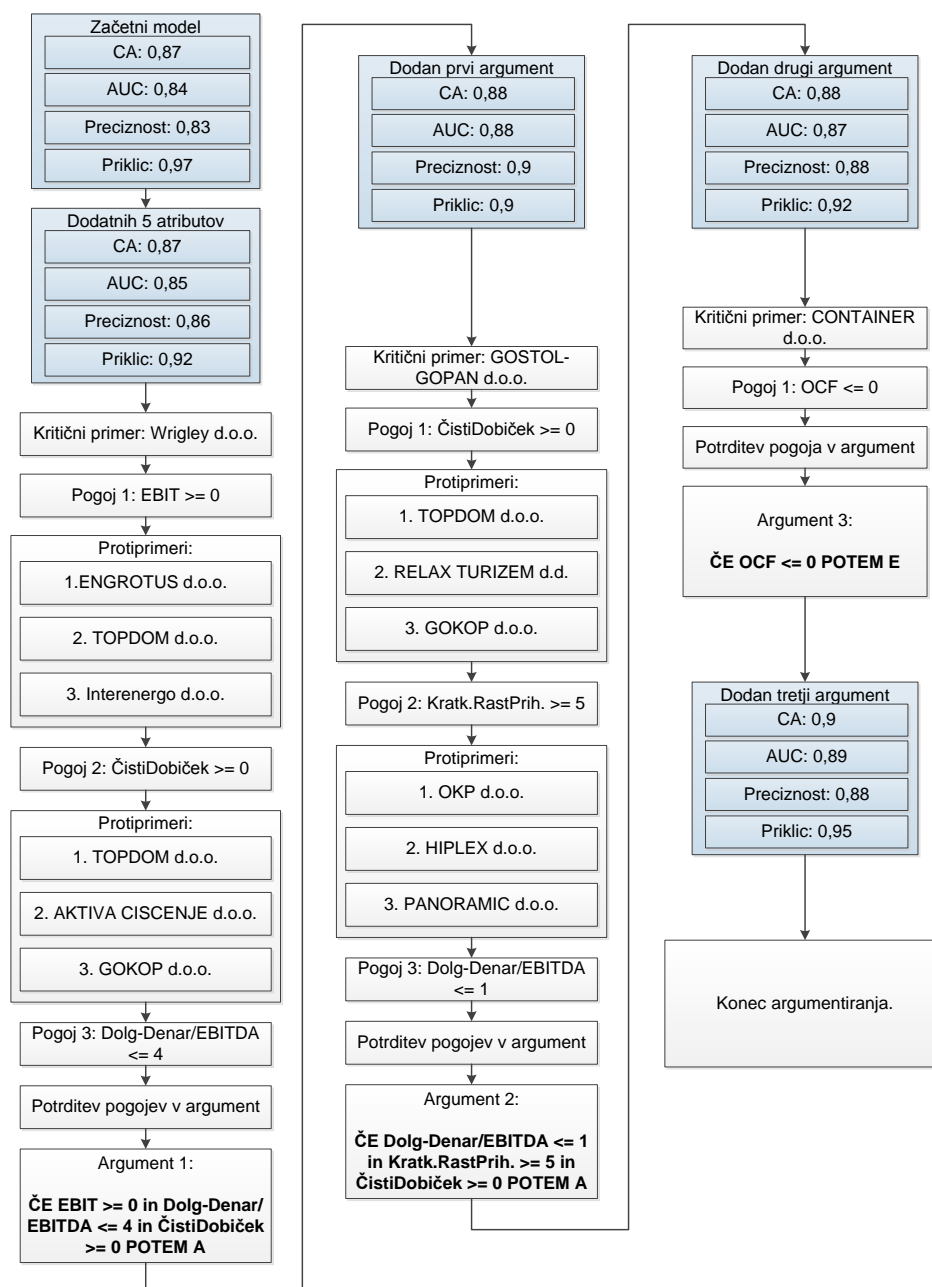
6.3.4 Eksperiment

Za ekspertiment smo uporabili podatke bonitetnih ocen podjetij, opisane v prejšnjem poglavju 6.3.2, in jih predstavili ekspertu. Ekspert je pred začetkom argumentiranja izdelal nove atribute, ki so mu koristili pri argumentiranju kritičnih primerov. Vsi uporabljeni atributi v eksperimentu so predstavljeni v tabeli 6.3. S krepkim besedilo so poudarjeni novo izdelani atributi.

Tabela 6.3: Ocene atributov

Ocena	Ime atributa	Ocena	Ime atributa	Ocena	Ime atributa
0.12 –	EBIT/Sredstva	0.1 –	Kapital/Sredstva	0.07 –	ČistiDobiček
0.06 –	FFO	0.05 –	Krat.Sred./Krat.Obv.	0.05 –	EBITDA
0.05 –	Dolg-Denar/EBITDA	0.04 –	EBIT	0.03 –	StroškiObresti
0.03 –	Dolg.RastPrihodkov	0.03 –	Kapital	0.03 –	StroškiDela
0.03 –	Krat.RastPrihodkov	0.03 –	Dolg.Sredstva	0.02 –	OdpisiVrednosti
0.02 –	Dolg	0.02 –	Krat.Pos.Obv.	0.02 –	EBIT/StroškiObr.
0.02 –	Krat.Obv.	0.02 –	OCF	0.02 –	Dolg.Obv.
0.02 –	CelotnePos.Obv.	0.01 –	Zaloge	0.01 –	Sredstva
0.01 –	FinančniOdhodki	0.01 –	Krat.Sredstva	0.01 –	Prihodki
0.01 –	StroskiBlagaInStoritev	0.01 –	Denar		

Prikaz celotnega postopka argumentiranja je prikazan na diagramu poteka na sliki 6.7.



Slika 6.7: Diagram poteka argumentiranja.

Kot prvi kritični primer se je prikazalo podjetje *Wrigley d.o.o.*, tipični predstavnik pa je postalo podjetje *Maros d.o.o.*, prikazana v tabeli 6.4. Podjetje *Wrigley d.o.o.* bi moralo biti uvrščeno v razred *A*, vendar ga je model logistične regresije klasificiral v razred *E*. Ekspert je podal naslednje opazke: *Podjetje Wrigley d.o.o. posluje pozitivno, ima visoke prihodke, dobiček in EBIT ter nima dolga.* Ekspert se po premisleku odloči za pogoj: $EBIT \geq 0$.

Tabela 6.4: Prvi kritični primer in tipični predstavnik

Atribut	Kritični primer	Tipični predstavnik
Izhodni razred	A	A
Podjetje	WRIGLEY d.o.o.	MAROS d.o.o.
Prihodki	36.966.304	2.435.931
StroskiBlagaInStoritev	19.695.908	1.844.581
StroskiDela	4.150.409	361.413
OdpisiVrednosti	255.228	179.350
FinančniOdhodki	18.349	20.089
StroskiObresti	18.349	20.089
EBIT	11.879.660	-2.659
EBITDA	12.134.888	176.691
ČistiDobiček	10.603.552	-17.144
Sredstva	23.176.082	6.619.956
Kapital	20.147.890	6.022.214
Dolg	0	0
Denar	1.010.546	258.001
Dolg.Sredstva	14.431.497	4.500.938
Krat.Sredstva	8.699.404	2.113.225
CelotnePos.Obv.	1.966.933	597.742
Krat.Pos.Obv.	1.966.933	465.002
Dolg.Obv.	0	132.740
Krat.Obveznosti	1.966.933	465.002
Zaloge	555.055	986.416
FFO	10.803.734	162.206
OCF	11.291.764	175.695
Dolg.RastPrihodkov	-5.5	-20.3
Krat.RastPrihodkov	-5.4	-9.7
Krat.Sred./Krat.Obv.	4.42	4.54
Dolg-Denar/EBITDA	-0.08	-1.46
Kapital/Sredstva	0.87	0.91
EBIT/StroskiObresti	647.43	-0.13
EBIT/Sredstva	0.51	-0.0

Glede na pogoj, ki ga je ekspert postavil, dobimo protiprimere, prikazane v tabeli 6.5. Iz danih protiprimerov ekspert opazi, da imata dve podjetji negativen čisti dobiček, kar pomeni, da podjetje slabo posluje oz. dela izgubo. Postavi nov pogoj: $\check{C}istiDobi\check{c}ek \geq 0$.

Tabela 6.5: Protiprimeri na podan pogoj $EBIT \geq 0$

Atribut	Kritični primer	Protiprimer 1	Protiprimer 2	Protiprimer 3
Izhodni razred	A	E	E	E
Podjetje	WRIGLEY d.o.o.	ENGROTUS d.o.o.	TOPDOM d.o.o.	Interenergo d.o.o.
Prihodki	36.966.304	509.149.792	71.615.440	167.241.344
StroskiBlagaInStor.	19.695.908	457.052.128	68.707.856	165.616.496
StroskiDela	4.150.409	44.705.280	2.041.815	1.170.110
OdpisiVred.	255.228	5.940.661	550.384	212.915
FinančniOdhodki	18.349	54.740.556	300.327	3.819.971
StroskiObresti	18.349	4.309.958	300.327	0.0
EBIT	11.879.660	1.472.124	292.047	409.236
EBITDA	12.134.888	7.412.785	842.431	622.151
ČistiDobiček	10.603.552	-48.991.708	231.415	-2.202.476
Sredstva	23.176.082	281.361.472	25.015.678	45.547.696
Kapital	20.147.890	39.351.704	7.621.415	15.371.581
Dolg	0.0	101.721.200	8.210.098	17.650.658
Denar	1.010.546	6.964.930	269.669	429.504
Dolg.Sredstva	14.431.497	179.654.416	10.639.961	30.538.202
Krat.Sredstva	8.699.404	99.573.920	14.351.838	2.540.040
CelotnePos.Obv.	1.966.933	132.458.360	9.171.931	6.612.922
Krat.Pos.Obv.	1.966.933	132.455.192	9.171.931	6.603.988
Dolg.Obv.	0.0	94.666.480	6.084.633	15.958.934
Krat.Obv.	1.966.933	139.619.168	11.297.396	8.304.646
Zaloge	555.055	39.621.340	3.088.244	0.0
FFO	10.803.734	-44.436.884	756.171	-2.009.817
OCF	11.291.764	-42.462.508	-276.148	-3.402.488
Dolg.RastPrihodkov	-5.5	-8.6	-5.0	39.5
Krat.RastPrihodkov	-5.4	-9.6	-3.5	-15.0
Krat.Sredstva/Krat.Obv.	4.42	0.71	1.27	0.31
Dolg-Denar/EBITDA	-0.08	12.78	9.43	27.68
Kapital/Sredstva	0.87	0.14	0.3	0.34
EBIT/StroskiObresti	647.43	0.34	0.97	0.0
EBIT/Sredstva	0.51	0.01	0.01	0.01

Prikažejo se novi protiprimeri (glej tabelo 6.6). Opazimo, da se med protiprimeri zopet pojavi podjetje *TOPDOM d.o.o.*, ki še vedno zadovoljuje vsem pogojem. Po pregledu kritičnega primera in protiprimerov ekspert opazi, da imajo vsa podjetja visok finančni kazalnik za odplačilo dolga (Dolg-Denar/EBITDA). Kazalnik meri sposobnost podjetja, da na dolgi rok poravnava nastali dolg. Vrednosti ≥ 4 kažejo na večje tveganje odplačila dolga v prihodnosti. Ekspert se odloči za dodaten pogoj: $Dolg-Denar/EBITDA \leq 4$.

Tabela 6.6: Protiprimeri na podana pogoja $EBIT \geq 0$ in $\check{C}istiDobi\check{c}ek \geq 0$

Atribut	Kritični primer	Protiprimer 1	Protiprimer 2	Protiprimer 3
Izhodni razred	A	E	E	E
Podjetje	WRIGLEY d.o.o.	TOPDOM d.o.o.	AKTIVA CISCENJE d.o.o.	GOKOP d.o.o.
Prihodki	36.966.304	71.615.440	20.878.784	6.086.919
StroskiBlagaInStor.	19.695.908	68.707.856	10.096.700	4.912.799
StroskiDela	4.150.409	2.041.815	10.235.583	358.768
OdpisiVred.	255.228	550.384	357.736	515.828
FinančniOdhodki	18.349	300.327	293.897	153.299
StroskiObresti	18.349	300.327	293.897	153.299
EBIT	11.879.660	292.047	193.874	200.939
EBITDA	12.134.888	842.431	551.610	716.767
ČistiDobiček	10.603.552	231.415	257.794	71.844
Sredstva	23.176.082	25.015.678	11.671.737	9.739.328
Kapital	20.147.890	7.621.415	3.705.541	1.903.544
Dolg	0.0	8.210.098	4.115.679	4.105.160
Denar	1.010.546	269.669	7.978	20.298
Dolg.Sredstva	14.431.497	10.639.961	5.115.535	2.601.389
Krat.Sredstva	8.699.404	14.351.838	6.545.104	7.137.670
CelotnePos.Obv.	1.966.933	9.171.931	2.811.441	3.427.178
Krat.Pos.Obv.	1.966.933	9.171.931	2.811.441	3.427.178
Dolg.Obv.	0.0	6.084.633	1.011.121	3.301.163
Krat.Obv.	1.966.933	11.297.396	5.915.999	4.231.175
Zaloge	555.055	3.088.244	54.724	125.038
FFO	10.803.734	756.171	577.344	427.044
OCF	11.291.764	-276.148	1.079.531	56.670
Dolg.RastPrihodkov	-5.5	-5.0	-7.6	-13.1
Krat.RastPrihodkov	-5.4	-3.5	-5.2	-46.2
Krat.Sredstva/Krat.Obv.	4.42	1.27	1.11	1.69
Dolg-Denar/EBITDA	-0.08	9.43	7.45	5.7
Kapital/Sredstva	0.87	0.3	0.32	0.2
EBIT/StroskiObresti	647.43	0.97	0.66	1.31
EBIT/Sredstva	0.51	0.01	0.02	0.02
EBIT/Sredstva	0.51	0.01	0.01	0.01

Prikažeta se dva protiprimera. Ker je ekspert zadovoljen s podanimi pogoji in bi rad končal argumentiranje primera, pogoje potrdi v argument. Izračuna se ocena argumenta prikazana v tabeli 6.7, argument pa se kot nov atribut doda v model. V tabeli opazimo, da je ocena argumenta zelo visoka. Argumenti, ki imajo oceno nad 0,9, veljajo za dobre argumente.

Tabela 6.7: Ocena prvega argumenta

Ocena argument	Argument
[0.93]	ČE $EBIT \geq 0$ in $Dolg-Denar/EBITDA \leq 4$ in ČistiDobiček ≥ 0 POTEM A

Model se ponovno izračuna in ekspertu ponudi nov kritični primer in tipičnega predstavnika, predstavljena v tabeli 6.8.

Ekspert opazi, da je pri tipičnem predstavniku lahko nekaj narobe in argumentira: *Podjetje MAROS d.o.o. ima negativen EBIT in negativen čisti dobiček in dela izgubo. Opazimo tudi, da je finančni kazalnik Dolg-Denar/EBITDA negativen. Razlog za to je dolg, ki je enak nič. Ta podatek bi se moral upoštevati le v primeru, da dolg ni enak nič.* Po premisleku se odloči za naslednji pogoj: $\check{C}istiDobi\check{c}ek \geq 0$. Pogoj sicer ni skladen s tipičnim predstavnikom razreda, odraža pa domensko znanje eksperta.

Tabela 6.8: Tipični predstavnik in kritični primer po prvem argumentu

Atribut	Kritični primer	Tipični predstavnik
Izhodni razred	A	A
Podjetje	GOSTOL-GOPAN d.o.o.	MAROS d.o.o.
Prihodki	15.146.377	2.435.931
StroskiBlagaInStor.	10.137.179	1.844.581
StroskiDela	3.876.224	361.413
OdpisiVred.	562.796	179.350
FinančniOdhodki	227.012	20.089
StroskiObresti	227.012	20.089
EBIT	634.964	-2.659
EBITDA	1.197.760	176.691
ČistiDobiček	437.518	-17.144
Sredstva	10.515.065	6.619.956
Kapital	2.382.394	6.022.214
Dolg	2.156.655.0	0
Denar	1.897.482	258.001
Dolg.Sredstva	4.032.444	4.500.938
Krat.Sredstva	6.458.675	2.113.225
CelotnePos.Obv.	5.036.867	597.742
Krat.Pos.Obv.	3.774.174	465.002
Dolg.Obv.	3.419.348	132.740
Krat.Obv.	3.774.174	465.002
Zaloge	2.384.903	986.416
FFO	688.731	162.206
OCF	1.955.594	175.695
Dolg.RastPrihodkov	-2.9	-20.3
Krat.RastPrihodkov	32.0	-9.7
Krat.Sredstva/Krat.Obv.	1.71	4.54
Dolg-Denar/EBITDA	0.22	-1.46
Kapital/Sredstva	0.23	0.91
EBIT/StroskiObresti	2.8	-0.13
EBIT/Sredstva	0.06	-0.0

Prikažejo se novi kritični primeri. Vsi protiprimeri imajo nizko vrednost atributa *Kratk.RastPrihodkov*. Ekspert se odloči za pogoj: *Krat.RastPrihodkov* ≥ 5 .

Tabela 6.9: Protiprimeri na podan pogoj $\check{C}istiDobi\check{c}ek \geq 0$

Atribut	Kritični primer	Protiprimer 1	Protiprimer 2	Protiprimer 3
Izhodni razred	A	E	E	E
Podjetje	GOSTOL-GOPAN d.o.o.	TOPDOM d.o.o.	RELAX TURIZEM d.d.	GOKOP d.o.o.
Prihodki	15.146.377	71.615.440	27.241.374	6.086.919
StroskiBlagaInStor.	10.137.179	68.707.856	25.386.414	4.912.799
StroškiDela	3.876.224	2.041.815	1.075.860	358.768
OdpisiVred.	562.796	550.384	277.509	515.828
FinančniOdhodki	227.012	300.327	212.683	153.299
StroškiObresti	227.012	300.327	209.099	153.299
EBIT	634.964	292.047	513.366	200.939
EBITDA	1.197.760	842.431	790.875	716.767
ČistiDobiček	437.518	231.415	307.275	71.844
Sredstva	10.515.065	25.015.678	9.389.734	9.739.328
Kapital	2.382.394	7.621.415	1.424.958	1.903.544
Dolg	2.156.655	8.210.098	3.735.980	4.105.160
Denar	1.897.482	269.669	90.841	20.298
Dolg.Sredstva	4.032.444	10.639.961	4.141.465	2.601.389
Krat.Sredstva	6.458.675	14.351.838	4.813.512	7.137.670
CelotnePos.Obv.	5.036.867	9.171.931	4.226.746	3.427.178
Krat.Pos.Obv.	3.774.174	9.171.931	4.226.746	3.427.178
Dolg.Obv.	3.419.348	6.084.633	0.0	3.301.163
Krat.Obv.	3.774.174	11.297.396	7.962.726	4.231.175
Zaloge	2.384.903	3.088.244	0.0	125.038
FFO	688.731	756.171	544.589	427.044
OCF	1.955.594	-276.148	1.639.490	56.670
Dolg.RastPrihodkov	-2.9	-5.0	5.6	-13.1
Krat.RastPrihodkov	32.0	-3.5	1.1	-46.2
Krat.Sredstva/Krat.Obv.	1.71	1.27	0.6	1.69
Dolg-Denar/EBITDA	0.22	9.43	4.61	5.7
Kapital/Sredstva	0.23	0.3	0.15	0.2
EBIT/StroškiObresti	2.8	0.97	2.46	1.31
EBIT/Sredstva	0.06	0.01	0.05	0.02

V protiprimerih v tabeli 6.10 ekspert zopet opazi visoko vrednost kazalca $Dolg-Denar/EBITDA$, kar je razlog, da se odloči za pogoj: $Dolg-Denar/EBITDA \leq 1$. Za vrednost 1 se odloči z razlogom, da pogoj ne bo enak pogoju v prejšnjem argumentu, hkrati pa bo pokril kritični primer in vse protiprimere. Metoda ne vrne nobenega protiprimera, zato pogoje potrdimo v argument.

Tabela 6.10: Protiprimeri na podana pogoja $\check{C}istiDobi\check{c}ek \geq 0$ in $Krat.RastPrihodkov \geq 5$

Atribut	Kritični primer	Protiprimer 1	Protiprimer 2	Protiprimer 3
Izhodni razred	A	E	E	E
Podjetje	GOSTOL-GOPAN d.o.o.	OKP d.o.o.	HIPLEX d.o.o.	PANORAMIC d.o.o.
Prihodki	15.146.377	6.369.930	2.559.303	5.031.143
StroskiBlagaInStor.	10.137.179	4.762.897	2.110.660	4.457.411
StroskiDela	3.876.224	1.830.351	332.359	245.934
OdpisiVred.	562.796	155.761	121.725	152.202
FinančniOdhodki	227.012	9.453	37.341	101.709
StroskiObresti	227.012	9.453	37.341	101.709
EBIT	634.964	-125.151	32.380	172.321
EBITDA	1.197.760	30.610	154.105	324.523
ČistiDobiček	437.518	4.409	5.474	47.913
Sredstva	10.515.065	2.980.080	1.974.999	3.735.187
Kapital	2.382.394	184.971	281.900	356.138
Dolg	2.156.655	573.362	937.511	1.400.901
Denar	1.897.482	218.457	24.969	41.723
Dolg.Sredstva	4.032.444	1.104.769	997.002	1.152.475
Krat.Sredstva	6.458.675	1.873.283	973.454	2.514.669
CelotnePos.Obv.	5.036.867	1.803.922	755.588	1.978.148
Krat.Pos.Obv.	3.774.174	1.803.922	755.588	1.978.148
Dolg.Obv.	3.419.348	511.402	247.511	747.352
Krat.Obv.	3.774.174	1.865.882	1.445.588	2.631.697
Zaloge	2.384.903	48.239	480.453	0.0
FFO	688.731	105.848	127.199	155.102
OCF	1.955.594	5.694	-39.650	593.933
Dolg.RastPrihodkov	-2.9	2.0	17.5	6.5
Krat.RastPrihodkov	32.0	17.9	8.3	35.7
Krat.Sredstva/Krat.Obv.	1.71	1.0	0.67	0.96
Dolg-Denar/EBITDA	0.22	11.59	5.92	4.19
Kapital/Sredstva	0.23	0.06	0.14	0.1
EBIT/StroskiObresti	2.8	-13.24	0.87	1.69
EBIT/Sredstva	0.06	-0.04	0.02	0.05

Dobimo nov argument, ki ga prikažemo v tabeli 6.11 skupaj s prejšnjim argumentom. Opazimo, da je nov argument dobil zelo slabo oceno.

Tabela 6.11: Ocene dosedanjih argumentov

Ocena argument	Argument
[0.93]	ČE $EBIT \geq 0$ in $Dolg-Denar/EBITDA \leq 4$ in $\check{C}istiDobiček \geq 0$ POTEM A
[0.2]	ČE $Dolg-Denar/EBITDA \leq 1$ in $Krat.RastPrih. \geq 5$ in $\check{C}istiDob. \geq 0$ POTEM A

Po potrditvi novega argumenta se zopet prikaže kritični primer. Ekspert se odloči, da bi rad argumentiral primer, ki bi se moral uvrstiti v razred bonitetne ocene E, vendar ga je metoda klasificirala kot razred A. Kritični primer zato zamenja. Pojavi se naslednji problematičen primer, prikazan v tabeli 6.12. Po pregledu podatkov ekspert opazi negativen kazalnik OCF , ki pove, da podjetje nima zadostnih prilivov na račun. Iz tega vemo, da podjetje ni sposobno sproti plačevati obveznosti, kar je razlog, da kritični primer sodi v razred E. Ekspert se odloči za pogoj: $OCF \leq 0$.

Tabela 6.12: Kritični primer po dveh argumentih

Atribut	Kritični primer	Tipični predstavnik
Izhodni razred	E	E
Podjetje	CONTAINER d.o.o.	IGM ZAGORJE d.o.o.
Prihodki	11.599.052	11.067.590
StroskiBlagaInStor.	10.573.720	9.196.540
StroskiDela	1.964.185	1.207.635
OdpisiVred.	20.917	841.335
FinančniOdhodki	7.369	64.270
StroskiObresti	7.369	52.861
EBIT	34.689	-183.113
EBITDA	55.606	658.222
ČistiDobiček	34.295	-226.270
Sredstva	5.340.145	11.315.742
Kapital	940.430	5.774.055
Dolg	0	2.754.088
Denar	3.558	534.133
Dolg.Sredstva	29.843	7.121.754
Krat.Sredstva	5.301.185	4.183.292
CelotnePos.Obv.	3.492.064	2.575.624
Krat.Pos.Obv.	3.492.064	2.575.624
Dolg.Obv.	0	1.761.531
Krat.Obv.	3.492.064	3.568.181
Zaloge	1.941.394	375.803
FFO	55.212	575.890
OCF	-229.909	922.779
Dolg.RastPrihodkov	8.0	-0.9
Krat.RastPrihodkov	-4.4	6.2
Krat.Sredstva/Krat.Obv.	1.52	1.17
Dolg-Denar/EBITDA	-0.06	3.37
Kapital/Sredstva	0.18	0.51
EBIT/StroskiObresti	4.71	-3.46
EBIT/Sredstva	0.01	-0.02

Pojavi se le en protiprimer, kar je razlog, da je ekspert zadovoljen z enostavnim pogojem in pogoj potrdi v argument. Argument prikažemo v tabeli 6.13 z obstoječimi argumenti. Iz tabele opazimo, da je argument dobil visoko oceno. Ker je ekspert z argumentiranjem zadovoljen, argumentiranje zaključimo. Rezultate argumentiranja in spremembe mer za evalvacijo prikažemo v tabeli 6.14.

Tabela 6.13: Ocene argumentov

Ocena argument	Argument
[0.93]	$\checkmark \text{E EBIT} \geq 0$ in $\text{Dolg-Denar/EBITDA} \leq 4$ in $\checkmark \text{istiDobiček} \geq 0$ POTEM A
[0.20]	$\checkmark \text{E Dolg-Denar/EBITDA} \leq 1$ in $\text{Krat.RastPrih.} \geq 5$ in $\checkmark \text{istiDob.} \geq 0$ POTEM A
[0.89]	$\checkmark \text{E OCF} \leq 0$ POTEM E

V tabeli rezultatov opazimo, da je z argumentiranjem mogoče izboljšati napovedni model, hkrati pa je napovedni model bolj skladen z domenskim znanjem. V iteraciji 1 je začetni model narobe klasificiral 9 testnih primerov. Po končanem argumentiranju pa je model narobe klasificiral le še 7 primerov. Opazimo tudi, da v primeru, ko dodamo slabo ocenjen argument, modela bistveno ne poslabšamo. To opazimo tudi pri iteraciji 6, kjer slab argument odstranimo iz modela. Ker smo bili omejeni z majhno množico podatkov, so lahko rezultati delno povezani z naključji. Dobrodošel bi bil test na večji testni množici podatkov oz. na množici, kjer bi na začetku imeli manjšo napovedno točnost modela.

Tabela 6.14: Napovedane točnosti modela

Iteracija	Nap. Toč.	AUC	Preciznost	Priklic	Št. Nap. Klas.	Komentar
1	0,87	0,84	0,83	0,97	9	Začetni model z obstoječimi atributi.
2	0,87	0,85	0,86	0,92	9	Dodatnih 5 atributov.
3	0,88	0,88	0,9	0,9	8	Dodan prvi argument.
4	0,88	0,87	0,88	0,92	8	Dodan drugi argument.
5	0,9	0,89	0,88	0,95	7	Dodan tretji argument.
6	0,9	0,89	0,9	0,92	7	Izbris drugega argumenta.

Poglavje 7

Sklepne ugotovitve

V magistrski nalogi smo predstavili eno izmed možnosti dodajanja ekspertnega znanja v metodo strojnega učenja, ki jo poznamo pod imenom logistična regresija. Vzeli smo najbolj osnovni model logistične regresije, ki ima dva razreda in omogoča binarno klasifikacijo. V okviru paradigme argumentiranega strojnega učenja (*ABML*) smo modelu logistične regresije dodali možnost zajema domenskega znanja in razvili novo metodo logistične regresije, ki omogoča interakcijo med domenskim ekspertom in strojnim učenjem z uporabo logistične regresije (angl. *argument-based machine learning with logistic regression*).

Nova metoda poišče napačno uvrščene oz. problematične primere, ki jih strojno učenje z logistično regresijo uvrsti v napačen razred in jih ponudi ekspertu v razlago. Ekspert s podajanjem argumentov v interaktivni zanki podaja razloge, čemu se določeni primeri uvrstijo v napačen razred. Glede na podane pogoje eksperta se poiščejo protiprimeri, ki lahko kažejo na pomanjkljivosti v pogojih. Ekspertu je omogočeno, da pogoje glede na protiprimerje in v skladu s svojim ekspertnim znanjem tudi izboljšuje. Ko je ekspert zadovoljen s podanimi pogoji, ima možnost te pogoje spremeniti v argument. Z izdelavo argumenta se pravilo, ki določa izbrani razred glede na podane pogoje, v obliki novega atributa doda v model. Metoda strojnega učenja ponovno izračuna hipotezo in poišče napačno uvrščene primere. Ekspertu je

omogočeno, da v kateremkoli trenutku preneha z argumentiranjem, s tem pa se lahko izogne pretiranemu prilagajanju podatkov. K slednjemu pripomore tudi to, da se ekspert pri uporabi argumentov neprestano opira na svoje znanje o domeni.

Novo razvita metoda vodi do napovednih modelov, ki so običajno bolj skladni z domenskim znanjem eksperta in ki praviloma vodijo do izboljšanja napovedne točnosti. Za lažjo uporabo novo izdelane metode smo razvili aplikacijo, ki uporablja opisani pristop, hkrati pa lahko služi tudi kot orodje za spoznanje nove domene podatkov. Aplikacija je prosto dostopna na spletu.

V rezultatih smo pokazali, da izdelava novih atributov pripomore k boljšim napovednim modelom. Pokazali smo, da novi atributi ne le izboljšajo napovedno točnost logistične regresije, pač pa tudi drugih metod strojnega učenja, ki smo jih uporabili v eksperimentu na domeni *Pima Indians*. V eksperimentu, kjer smo v model preko interaktivne zanke za zajem domenskega znanja vnašali znanje domenskega strokovnjaka, smo pokazali, da dobri argumenti pripomorejo k izboljšanim napovednim točnostim modela. V končnem modelu smo uporabili tudi naprednejše attribute, ki jih domenski eksperti tipično uporabljajo pri ocenjevanju tveganj pri podeljevanju kreditov. Ti napredni atributi so bili tipično bolje ocenjeni od atributov v osnovni množici podatkov.

Kot eno izmed možnih dodatnih izboljšav bi omenili, da bi obstoječi metodi lahko dodali možnost samodejnega iskanja čim ustreznejših mej za podane argumente. Druga možna izboljšava, ki se nanaša zlasti na izboljšanje uporabniške izkušnje, bi bila izdelava vtičnika v okolju *Orange*, kar bi olajšalo pripravo podatkov in njihovo vizualizacijo. V poštev pride tudi implementacija novo razvite metode v obliki spletne aplikacije in nadgradnje spletne aplikacije v inteligentni sistem, namenjen poučevanju. Pri tem naj poudarimo, da je predstavljeni pristop mogoče uporabiti tako rekoč v katerikoli domeni, ki jo je moč predstaviti kot klasifikacijski problem.

Izboljšave so možne tudi na nivoju samega napovednega modela, kjer bi zamenjali osnovno metodo logistične regresije s katero drugo. V prvi vrsti

bi lahko dodali možnost, da logistična regresija sprejme več kot dva razreda (angl. *multinomial logistic regression*). V sorodnih delih smo omenili lokalno uteženo logistično regresijo, ki bi ji lahko dodali možnost argumentiranja in vnosa ekspertnega znanja.

Literatura

- [1] E. Alpaydin, Introduction to Machine Learning, The MIT Press Cambridge, Massachusetts London, England, 2010.
- [2] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal, Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann, 2011.
- [3] J. F. Ameet Talwalkar, Jon C. Bates, BerkeleyX: CS120x Distributed Machine Learning with Apache Spark.
- [4] J. Demšar, T. Curk, A. Erjavec, Črt Gorup, T. Hočevár, M. Milutinovič, M. Možina, M. Polajnar, M. Toplak, A. Starič, M. Štajdohar, L. Umek, L. Žagar, J. Žbontar, M. Žitnik, B. Zupan, Orange: Data mining toolbox in python, Journal of Machine Learning Research 14 (2013) 2349–2353.
- [5] M. Možina, Argument based machine learning (doctoral dissertation), University of Ljubljana, 2009.
- [6] P. Šaponja, Clustering with argument-based machine learning (master's thesis), University of Ljubljana, 2015.
- [7] M. Pavlič, Estimating the quality of arguments in argument-based machine learning (master's thesis), University of Ljubljana, 2015.
- [8] P.-N. Tan, et al., Introduction to data mining, Pearson Education India, 2006.
- [9] A. Ng, Machine Learning by Stanford University.

-
- [10] C. M. Bishop, Pattern recognition and machine learning, Springer, 2006.
 - [11] N. Landwehr, M. Hall, E. Frank, Logistic model trees, Machine Learning 59 (1-2) (2005) 161–205.
 - [12] K. Deng, Omega: On-line memory-based general purpose system classifier, Ph.D. thesis, Georgia Institute of Technology (1998).
 - [13] P. Domingos, A few useful things to know about machine learning, Communications of the ACM 55 (10) (2012) 78–87.
 - [14] H.-P. H. Hsiang-Fu Yu, Hung-Yi Lo, Feature engineering and classifier ensemble for kdd cup 2010, Vol. 1, JMLR: Workshop and Conference Proceedings, 2010, pp. 1–16.
 - [15] M. Možina, M. Guid, J. Krivec, A. Sadikov, I. Bratko, Learning to explain with ABML., in: ExaCt, 2010, pp. 37–48.
 - [16] M. Možina, J. Žabkar, I. Bratko, Argument based machine learning, Artificial Intelligence 171 (10-15) (2007) 922–937.
 - [17] M. Guid, M. Možina, V. Groznik, D. Georgiev, A. Sadikov, Z. Pirtošek, I. Bratko, ABML knowledge refinement loop: A case study, in: International Symposium on Methodologies for Intelligent Systems, Springer, 2012, pp. 41–50.
 - [18] M. Možina, M. Guid, J. Krivec, A. Sadikov, I. Bratko, Fighting knowledge acquisition bottleneck with argument based machine learning., in: ECAI, 2008, pp. 234–238.
 - [19] M. Zapušek, M. Možina, I. Bratko, J. Rugelj, M. Guid, Designing an interactive teaching tool with ABML knowledge refinement loop, in: International Conference on Intelligent Tutoring Systems, Springer, 2014, pp. 575–582.

-
- [20] M. Guid, M. Možina, J. Krivec, A. Sadikov, I. Bratko, Learning positional features for annotating chess games: A case study, in: International Conference on Computers and Games, Springer, 2008, pp. 192–204.
- [21] V. Groznik, M. Guid, A. Sadikov, M. Možina, D. Georgiev, V. Kragelj, S. Ribarič, Z. Pirtošek, I. Bratko, Elicitation of neurological knowledge with argument-based machine learning, *Artificial intelligence in medicine* 57 (2) (2013) 133–144.
- [22] K. Napierała, J. Stefanowski, Argument based generalization of modlem rule induction algorithm, in: International Conference on Rough Sets and Current Trends in Computing, Springer, 2010, pp. 138–147.
- [23] S. Džeroski, B. Cestnik, I. Petrovski, Using the m-estimate in rule induction, *Journal of Computing and Information Technology*, 1993, pp. 37–46.
- [24] Tkinter - grafični vmesnik, Dostopno na: <https://wiki.python.org/moin/TkInter>, (pridobljeno 05.01.2017) (2017).
- [25] D. Možina, M. Guid, Argument-based machine learning with logistic regression, Dostopno na: <https://ailab.si/ablr/>, (pridobljeno 29.01.2017) (2017).
- [26] M. Lichman, UCI machine learning repository (2013).
URL <http://archive.ics.uci.edu/ml>
- [27] Gvin - položaj podjetij na slovenskem trgu, Dostopno na: <http://www.bisnode.si/produkt/gvin/>, (pridobljeno 05.01.2017) (2017).
- [28] Poslovni asistent bizi.si, Dostopno na: <http://www.bizi.si/informacije/pomoc/>, (pridobljeno 05.01.2017) (2017).
- [29] Osnove računovodstva, Dostopno na: <http://skrci.me/aDUXA>, (pridobljeno 05.01.2017) (2017).